バージョン13

スクリプト構文リファレンス

「真の発見の旅とは、新しい風景を探すことではなく、新たな視点を持つことである。」
マルセル・プルースト

JMP, A Business Unit of SAS
SAS Campus Drive
Cary, NC 27513

13.1

このマニュアルを引用する場合は、次の正式表記を使用してください: SAS Institute Inc. 2017.
『JMP® 13 スクリプト構文リファレンス』Cary, NC: SAS Institute Inc.

JMP® 13 スクリプト構文リファレンス

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

2016年9月

2017年2月

SASソフトウェアは、オープンソースのソフトウェアを含むがそれに限らない、特定のサードパーティ製ソフトウェアと共に提供される場合があります。かかるソフトウェアは、適用されるサードパーティソフトウェアライセンス契約に基づいてライセンスを得たものです。SASソフトウェアと共に配布されるサードパーティ製ソフトウェアに関する情報は、http://support.sas.com/thirdpartylicenses を参照してください。

テクノロジーライセンスに関する通知

- Scintilla - Copyright © 1998-2014 by Neil Hodgson <neilh@scintilla.org>.

  All Rights Reserved.

  何らかの目的でこのソフトウェアとそのマニュアルを手数料なしで使用、コピー、変更および配布することは、これをもって許可されます。ただし、すべてのコピーに上記の著作権に関する通知が記載されていること、および補助的なマニュアルに著作権に関する通知とこの許可に関する通知の両方が記載されていることを条件とします。

  NEIL HODGSONは、商業性および適合性の黙示的な保証を含め、このソフトウェアに関するすべての保証を放棄します。NEIL HODGSONは、いかなる場合においても、それが契約、過失、もしくは他の不法行為のどれであれ、このソフトウェアの使用もしくは性能から生じた、もしくはそれに関連して生じた使用、データ、もしくは利益の損失の結果として生じる特別損害、間接損害、もしくは付随的損害を始めとするいかなる損害に対しても責任を負いません。

- Telerik RadControls: Copyright © 2002-2012, Telerik. 含まれているTelerik RadControlsをJMP以外で使用することは許可されていません。
- ZLIB圧縮ライブラリ - Copyright © 1995-2005, Jean-Loup Gailly and Mark Adler.
- Natural Earthを使用して作成。無料のベクトルおよびラスター地図データ @ naturalearthdata.com.
- パッケージ - Copyright © 2009-2010, Stéphane Sudre (s.sudre.free.fr). All rights reserved.

  ソースおよびバイナリの形で、そのまま、もしくは変更を加えて再配布および使用することは、次のような条件を満たす限り、許可されます。

  再配布するソースコードには、上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  バイナリ形式で再配布する場合は、共に提供されるマニュアルなどの資料に上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  事前に書面による許可を得ることなく、このソフトウェアから派生した製品の推奨または宣伝のためにWhiteBoxの名前やその貢献者の名前を使用することはできません。

  このソフトウェアは、著作権保有者および貢献者によって「現状のままで」提供され、商業性および特定の目的に対する適合性に関する黙示的な保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。いかなる場合においても、著作権保有者または貢献者は、損害の原因が何であれ、そして法的責任の根拠が何であれ、つまり、契約、厳格責任、不法行為（過失その他を含む）の

どれであれ、かかる損害の発生する可能性を事前に知らされていたとしても、このソフトウェアをどのように使用して生じた損害であれ、いかなる直接損害、間接損害、付随的損害、特別損害、懲罰的損害、もしくは結果損害（代替品または代替サービスの調達、使用機会、データもしくは利益の損失、業務の中断を含むがそれに限らない）に対しても責任を負いません。

- iODBC ソフトウェア - Copyright © 1995-2006, OpenLink Software Inc and Ke Jin (www.iodbc.org). All rights reserved.

  ソースおよびバイナリの形で、そのまま、もしくは変更を加えて再配布および使用することは、次のような条件を満たす限り、許可されます。

  – 再配布するソースコードには、上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  – バイナリ形式で再配布する場合は、共に提供されるマニュアルなどの資料に上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  – 事前に書面による許可を得ることなく、このソフトウェアから派生した製品の推奨または宣伝のために OpenLink Software Inc. の名前やその貢献者の名前を使用することはできません。

  このソフトウェアは、著作権保有者および貢献者によって「現状のままで」提供され、商業性および特定の目的に対する適合性に関する黙示的な保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。いかなる場合においても、OPENLINK または貢献者は、損害の原因が何であれ、そして法的責任の根拠が何であれ、つまり、契約、厳格責任、不法行為（過失その他を含む）のどれであれ、かかる損害の発生する可能性を事前に知らされていたとしても、このソフトウェアをどのように使用して生じた損害であれ、いかなる直接損害、間接損害、付随的損害、特別損害、懲罰的損害、もしくは結果損害（代替品または代替サービスの調達、使用機会、データもしくは利益の損失、業務の中断を含むがそれに限らない）に対しても責任を負いません。

- bzip2、関連ライブラリの「libbzip2」、およびすべてのマニュアル: Copyright © 1996-2010, Julian R Seward. All rights reserved.

  ソースおよびバイナリの形で、そのまま、もしくは変更を加えて再配布および使用することは、次のような条件を満たす限り、許可されます。

  再配布するソースコードには、上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  このソフトウェアの供給源は正しく表記しなければならず、使用者が元のソフトウェアを記述したと主張することはできません。ある製品の中でこのソフトウェアを使用する場合は、その製品のマニュアルに謝辞を記載してもらえるとありがたいですが、必須ではありません。

  ソースに変更を加えたバージョンには、その旨を明記しなければならず、元のソフトウェアとは違うものであることを明確にしてください。

事前に書面による許可を得ることなく、このソフトウェアから派生した製品の推奨または宣伝のために作成者の名前を使用することはできません。

このソフトウェアは、作成者によって「現状のままで」提供され、商業性および特定の目的に対する適合性に関する黙示的な保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。いかなる場合においても、作成者は、損害の原因が何であれ、そして法的責任の根拠が何であれ、つまり、契約、厳格責任、不法行為（過失その他を含む）のどれであれ、かかる損害の発生する可能性を事前に知らされていたとしても、このソフトウェアをどのように使用して生じた損害であれ、いかなる直接損害、間接損害、付随的損害、特別損害、懲罰的損害、もしくは結果損害（代替品または代替サービスの調達、使用機会、データもしくは利益の損失、業務の中断を含むがそれに限らない）に対しても責任を負いません。

- R ソフトウェア : Copyright © 1999-2012, R Foundation for Statistical Computing.
- MATLAB ソフトウェア : Copyright © 1984-2012, The MathWorks, Inc. 米国特許法および国際特許法によって保護されています。www.mathworks.com/patents を参照してください。MATLAB および Simulink は、The MathWorks, Inc. の登録商標です。他の商標は、www.mathworks.com/trademarks に一覧されています。他の製品名やブランド名は、それぞれの所有者の商標または登録商標である可能性があります。
- libopc: Copyright © 2011, Florian Reuter. All rights reserved.

  ソースおよびバイナリの形で、そのまま、もしくは変更を加えて再配布および使用することは、次のような条件を満たす限り、許可されます。

  – 再配布するソースコードには、上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  – バイナリ形式で再配布する場合は、共に提供されるマニュアルなどの資料に上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  – 事前に書面による許可を得ることなく、このソフトウェアから派生した製品の推奨または宣伝のために Florian Reuter の名前やその貢献者の名前を使用することはできません。

  このソフトウェアは、著作権保有者および貢献者によって「現状のままで」提供され、商業性および特定の目的に対する適合性に関する黙示的な保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。いかなる場合においても、著作権保有者または貢献者は、損害の原因が何であれ、そして法的責任の根拠が何であれ、つまり、契約、厳格責任、不法行為（過失その他を含む）のどれであれ、かかる損害の発生する可能性を事前に知らされていたとしても、このソフトウェアをどのように使用して生じた損害であれ、いかなる直接損害、間接損害、付随的損害、特別損害、懲罰的損害、もしくは結果損害（代替品または代替サービスの調達、使用機会、データもしくは利益の損失、業務の中断を含むがそれに限らない）に対しても責任を負いません。
- libxml2 - ソースコードに特に記載がある場合を除く（たとえば、使用しているライセンスは類似しているが、著作権の通知が異なる hash.c、list.c ファイルや trio ファイル）、すべてのファイル :

Copyright © 1998 - 2003 Daniel Veillard. All Rights Reserved.

これをもって、このソフトウェアのコピーと関連する文書ファイル（「本ソフトウェア」）を入手した人すべてに対し、無料で本ソフトウェアを使用、コピー、変更、マージ、パブリッシュ、配布、サブライセンスする、もしくはコピーを販売する権利を含むがそれに限定せず、本ソフトウェアを制限なく取り扱う権利、および本ソフトウェアの供給相手に対してそうすることを許可する権利が付与されます。ただし、以下の条件を満たさなければなりません。

上記の著作権に関する通知とこの許可に関する通知が、本ソフトウェアのコピーのすべてまたは大部分に記載されていること。

このソフトウェアは、 現状のままで」提供され、商業性および特定の目的に対する適合性、および非侵害の保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。DANIEL VEILLARDは、いかなる場合においても、それが契約、過失、もしくは他の不法行為のどれであれ、本ソフトウェアから、もしくは本ソフトウェアに関連して、または本ソフトウェアの使用もしくは他の取り扱いに関連して生じた申し立て、損害賠償もしくは他の義務に対し、責任を負いません。

この通知に含まれているものを除き、Daniel Veillardから事前に書面による許可を得ることなく、本ソフトウェアの広告、またはその他の手段による本ソフトウェアの販売、使用もしくは他の取り扱いの宣伝にDaniel Veillardの名前を使用することはできません。

- UNIXファイルに使用された解凍アルゴリズムについて:

  Copyright © 1985, 1986, 1992, 1993

  カリフォルニア大学評議員。All rights reserved.

  このソフトウェアは、評議員および貢献者によって「現状のままで」提供され、商業性および特定の目的に対する適合性に関する黙示的な保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。いかなる場合においても、評議員または貢献者は、損害の原因が何であれ、そして法的責任の根拠が何であれ、つまり、契約、厳格責任、不法行為（過失その他を含む）のどれであれ、かかる損害の発生する可能性を事前に知らされていたとしても、このソフトウェアをどのように使用して生じた損害であれ、いかなる直接損害、間接損害、付随的損害、特別損害、懲罰的損害、もしくは結果損害（代替品または代替サービスの調達、使用機会、データもしくは利益の損失、業務の中断を含むがそれに限らない）に対しても責任を負いません。

  1. 再配布するソースコードには、上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  2. バイナリ形式で再配布する場合は、共に提供されるマニュアルなどの資料に上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

  3. 事前に書面による許可を得ることなく、このソフトウェアから派生した製品の推奨または宣伝のために大学の名前や貢献者の名前を使用することはできません。

- Snowball - Copyright © 2001, Dr Martin Porter, Copyright © 2002, Richard Boulton.

  All rights reserved.

ソースおよびバイナリの形で、そのまま、もしくは変更を加えて再配布および使用することは、次のような条件を満たす限り、許可されます。

1. 再配布するソースコードには、上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

2. バイナリ形式で再配布する場合は、共に提供されるマニュアルなどの資料に上記の著作権に関する通知、この条件リスト、これに続く放棄声明が記載されていなければなりません。

3. 事前に書面による許可を得ることなく、このソフトウェアから派生した製品の推奨または宣伝のために著作権保有者の名前や貢献者の名前を使用することはできません。

このソフトウェアは、著作権保有者および貢献者によって「現状のままで」提供され、商業性および特定の目的に対する適合性に関する黙示的な保証を含むがそれに限らない、いかなる明示的もしくは黙示的な保証も行われません。いかなる場合においても、著作権保有者または貢献者は、損害の原因が何であれ、そして法的責任の根拠が何であれ、つまり、契約、厳格責任、不法行為（過失その他を含む）のどれであれ、かかる損害の発生する可能性を事前に知らされていたとしても、このソフトウェアをどのように使用して生じた損害であれ、いかなる直接損害、間接損害、付随的損害、特別損害、懲罰的損害、もしくは結果損害（代替品または代替サービスの調達、使用機会、データもしくは利益の損失、業務の中断を含むがそれに限らない）に対しても責任を負いません。

目次
スクリプト構文リファレンス

第 1 章

JMP の概要
マニュアルとその他のリソース

この章には以下の情報が記載されています。

- 本書の表記法
- JMP のマニュアル
- JMP ヘルプ
- その他のリソース
  - その他の JMP のドキュメンテーション
  - チュートリアル
  - 索引
  - Web リソース
  - テクニカルサポートのオプション

表記規則

マニュアルの内容と画面に表示される情報を対応付けるために、次のような表記規則を使っています。

- サンプルデータ名、列名、パス名、ファイル名、ファイル拡張子、およびフォルダ名は「」で囲んで表記しています。
- スクリプトのコードは`Lucida Sans Typewriter`フォントで表記しています。
- スクリプトコードの結果（ログに表示されるもの）は*`Lucida Sans Typewriter`*（斜体）フォントで表記し、先に示すコードよりインデントされています。
- クリックまたは選択する項目は [] で囲んで太字で表記しています。これには以下の項目があります。
  - ボタン
  - チェックボックス
  - コマンド
  - 選択可能なリスト項目
  - メニュー
  - オプション
  - タブ名
  - テキストボックス
- 次の項目の表記規則は下記のとおりです。
  - 重要な単語や句、JMPに固有の定義を持つ単語や句は太字または「」で囲んで表記
  - マニュアルのタイトルは『』で囲んで表記
  - 変数名は斜体で表記
  - スクリプトの出力は斜体で表記
- JMP Proのみの機能にはJMP Proアイコン JMP PRO がついています。JMP Proの機能の概要については https://www.jmp.com/ja_jp/software/predictive-analytics-software.html をご覧ください。

メモ: 特別な情報および制限事項には、この文のように「メモ」という見出しがついています。

ヒント: 役に立つ情報には「ヒント」という見出しがついています。

JMP のマニュアル

JMP では、PDF 形式のマニュアルが用意されています。

- PDF 版は［**ヘルプ**］>［**ドキュメンテーション**］メニューまたは JMP オンラインヘルプのフッタから開くことができます。
- 検索しやすいようにすべてのドキュメンテーションが 1 つの PDF ファイルにまとめられた『JMP ドキュメンテーションライブラリ』と呼ばれるファイルがあります。『JMP ドキュメンテーションライブラリ』の PDF ファイルは［**ヘルプ**］>［**ドキュメンテーション**］メニューから開くことができます。

JMP ドキュメンテーションライブラリ

以下の表は、JMP ライブラリに含まれている各ドキュメンテーションの目的および内容をまとめたものです。

マニュアル	目的	内容
『はじめての JMP』	JMP をあまりご存知ない方を対象とした入門ガイド	JMP の紹介と、データを作成および分析し始めるための情報
『JMP の使用法』	JMP のデータテーブルと、基本操作を理解する	一般的な JMP の概念と、データの読み込み、列プロパティの変更、データの並べ替え、SAS への接続など、JMP 全体にわたる機能の説明
『基本的な統計分析』	このマニュアルを見ながら、基本的な分析を行う	［分析］メニューからアクセスできる以下のプラットフォームの説明： • 一変量の分布 • 二変量の関係 • 表の作成 • テキストエクスプローラ ［分析］>［二変量の関係］で二変量、一元配置分散分析、分割表に対する分析を実行する方法の説明。ブートストラップを使用した標本分布の近似方法やシミュレーションの機能を使用したパラメトリックな標本再抽出の実行方法の説明も含まれています。

マニュアル	目的	内容
『グラフ機能』	データに合った理想的なグラフを見つける	［グラフ］メニューからアクセスできる以下のプラットフォームの説明： • グラフビルダー • 重ね合わせプロット • 三次元散布図 • 等高線図 • バブルプロット • パラレルプロット • セルプロット • ツリーマップ • 散布図行列 • 三角図 • チャート このマニュアルには背景マップやカスタムマップの作成方法も記載されています。
『プロファイル機能』	対話式のプロファイルツールの使い方を学ぶ。任意の応答曲面の断面を表示できるようになります。	［グラフ］メニューに表示されるすべてのプロファイルについて。誤差因子の分析が、ランダム入力を使用したシミュレーションの実行とともに含まれています。
『実験計画(DOE)』	実験の計画方法と適切な標本サイズの決定方法を学ぶ	［実験計画（DOE）］メニューと［分析］>［発展的なモデル］メニューの「発展的な実験計画モデル」に関するすべてのトピックについて。

マニュアル	目的	内容
『基本的な回帰モデル』	「モデルのあてはめ」プラットフォームとその多くの手法について学ぶ	［分析］メニューの「モデルのあてはめ」プラットフォームで使用できる、以下の手法の説明： • 標準最小2乗 • ステップワイズ • 一般化回帰 • 混合モデル • MANOVA • 対数線形 - 分散 • 名義ロジスティック • 順序ロジスティック • 一般化線形モデル

マニュアル	目的	内容
『予測モデルおよび発展的なモデル』	さらなるモデリング手法について学ぶ	［分析］>［予測モデル］メニューで使用できる以下のプラットフォームの説明： • モデル化ユーティリティ • ニューラル • パーティション • ブートストラップ森 • ブースティングツリー • K近傍法 • 単純Bayes • モデルの比較 • 計算式デポ ［分析］>［発展的なモデル］メニューで使用できる以下のプラットフォームの説明： • 曲線のあてはめ • 非線形回帰 • Gauss過程 • 時系列分析 • 対応のあるペア ［分析］>［スクリーニング］メニューで使用できる以下のプラットフォームの説明： • 応答のスクリーニング • 工程のスクリーニング • 説明変数のスクリーニング • アソシエーション分析 ［分析］>［発展的なモデル］>［発展的な実験計画モデル］で使用できるプラットフォームについては、『実験計画(DOE)』に説明があります。

マニュアル	目的	内容
『多変量分析』	複数の変数を同時に分析するための手法について理解を深める	［分析］>［多変量］メニューで使用できる以下のプラットフォームの説明： • 多変量の相関 • 主成分分析 • 判別分析 • PLS ［分析］>［クラスター分析］メニューで使用できる以下のプラットフォームの説明： • 階層型クラスター分析 • K Means クラスター分析 • 正規混合 • 潜在クラス分析 • 変数のクラスタリング
『品質と工程』	工程を評価し、向上させるためのツールについて理解を深める	［分析］>［品質と工程］メニューで使用できる以下のプラットフォームの説明： • 管理図ビルダーと個々の管理図 • 測定システム分析 • 計量値/計数値ゲージチャート • 工程能力 • パレート図 • 特性要因図

マニュアル	目的	内容
『信頼性/生存時間分析』	製品やシステムにおける信頼性を評価し、向上させる方法、および人や製品の生存時間データを分析する方法について学ぶ	［分析］>［信頼性/生存時間分析］メニューで使用できる以下のプラットフォームの説明： • 寿命の一変量 • 寿命の二変量 • 累積損傷 • 再生モデルによる分析 • 劣化分析と破壊劣化 • 信頼性予測 • 信頼性成長 • 信頼性ブロック図 • 修理可能システムのシミュレーション • 生存時間分析 • 生存時間(パラメトリック)のあてはめ • 比例ハザードのあてはめ
『消費者調査』	消費者選好を調査し、その洞察を使用してより良い製品やサービスを作成するための方法を学ぶ	［分析］>［消費者調査］メニューで使用できる以下のプラットフォームの説明： • カテゴリカル • 多重対応分析 • 多次元尺度構成 • 因子分析 • 選択モデル • MaxDiff • アップリフト • 項目分析
『スクリプトガイド』	パワフルなJMPスクリプト言語（JSL）の活用方法について学ぶ	スクリプトの作成やデバッグ、データテーブルの操作、ディスプレイボックスの構築、JMPアプリケーションの作成など。
『スクリプト構文リファレンス』	JSL関数、その引数、およびオブジェクトやディスプレイボックスに送信するメッセージについて理解を深める	JSLコマンドの構文、例、および注意書き。

メモ：［**ドキュメンテーション**］メニューでは、印刷可能な 2 つのリファレンスカードも用意されています。『**メニューカード**』は JMP のメニューをまとめた表で、『**クイックリファレンス**』は JMP のショートカットキーをまとめた表です。

JMP ヘルプ

JMP ヘルプは、一連のマニュアルの簡易版です。JMP のヘルプは、次のいくつかの方法で開くことができます。

- Windows では、F1 キーを押すとヘルプシステムウィンドウが開きます。
- データテーブルまたはレポートウィンドウの特定の部分のヘルプを表示します。［**ツール**］メニューからヘルプツール を選択した後、データテーブルやレポートウィンドウの任意の位置でクリックすると、その部分に関するヘルプが表示されます。
- JMP ウィンドウ内で［**ヘルプ**］ボタンをクリックします。
- Windows の場合、［ヘルプ］メニューの［**ヘルプの目次**］、［**ヘルプの検索**］、［**ヘルプの索引**］の各オプションを使用して、JMP ヘルプ内を検索し、目的の内容を表示します。Mac の場合、［**ヘルプ**］ > ［**JMP ヘルプ**］を選択します。

JMP を習得するためのその他のリソース

JMP のマニュアルと JMP ヘルプの他、次のリソースも JMP の学習に役立ちます。

- チュートリアル（「チュートリアル」（21 ページ）を参照）
- サンプルデータ（「サンプルデータテーブル」（22 ページ）を参照）
- 索引（「統計用語と JSL 用語の習得」（22 ページ）を参照）
- 使い方ヒント（「JMP を使用するためのヒント」（22 ページ）を参照）
- Web リソース（「JMP User Community」（23 ページ）を参照）
- 専門誌『JMPer Cable』（「JMPer Cable」（23 ページ）を参照）
- JMP に関する書籍（「JMP 関連書籍」（23 ページ）を参照）
- JMP スターター（「「JMP スターター」ウィンドウ」（24 ページ）を参照）
- 教育用リソース（「サンプルデータテーブル」（22 ページ）を参照）

チュートリアル

［**ヘルプ**］ > ［**チュートリアル**］を選択して、JMP のチュートリアルを表示できます。［**チュートリアル**］メニューの最初の項目は［**チュートリアルディレクトリ**］です。この項目を選択すると、すべてのチュートリアルをカテゴリ別に整理した新しいウィンドウが開きます。

JMPに慣れていない方は、まず［**初心者用チュートリアル**］を試してみてください。JMPのインターフェースおよび基本的な使用方法を学ぶことができます。

他のチュートリアルでは、実験の計画、標本平均と定数の比較など、JMPの具体的な活用法を学習できます。

サンプルデータテーブル

JMPのマニュアルで取り上げる例は、すべてサンプルデータを使用しています。サンプルデータディレクトリを開くには、［**ヘルプ**］>［**サンプルデータライブラリ**］を選択します。

サンプルデータテーブルを文字コード順に並べた一覧を表示する、またはカテゴリごとにサンプルデータを表示するには、［**ヘルプ**］>［**サンプルデータ**］を選択します。

サンプルデータテーブルは次のディレクトリにインストールされています。

Windowsの場合: C:¥Program Files¥SAS¥JMP¥13¥Samples¥Data

Macintoshの場合: \Library\Application Support\JMP\13\Samples\Data

JMP Proでは、サンプルデータが（JMPではなく）JMPPROディレクトリにインストールされています。シングルユーザーライセンス版のJMP（JMP シュリンクラップ）では、サンプルデータがJMPSWディレクトリにインストールされています。

サンプルデータの使用例を参照するには、［**ヘルプ**］>［**サンプルデータ**］を選択し、教育用セクションから検索してください。教育用リソースについては、http://jmp.com/tools にも情報があります。

統計用語とJSL用語の習得

［**ヘルプ**］メニューには、次の索引が用意されています。

統計の索引　統計用語が説明されています。

スクリプトの索引　JSL関数、オブジェクト、ディスプレイボックスに関する情報を検索できます。スクリプトの索引からサンプルスクリプトを編集して実行することもできます。

JMPを使用するためのヒント

JMPを最初に起動すると、「使い方ヒント」ウィンドウが表示されます。このウィンドウには、JMPを使う上でのヒントが表示されます。

「使い方ヒント」ウィンドウを表示しないようにするには、［**起動時にヒントを表示する**］のチェックを外します。再表示するには、［**ヘルプ**］>［**使い方ヒント**］を選択します。または、「環境設定」ウィンドウで非表示に設定することもできます。詳細については、『JMPの使用法』を参照してください。

ツールヒント

次のような項目の上にカーソルを置くと、その項目を説明するツールヒントが表示されます。

- メニューまたはツールバーのオプション
- グラフ内のラベル
- レポートウィンドウ内の結果（テキスト）（カーソルで円を描くと表示される）
- 「ホームウィンドウ」内のファイル名またはウィンドウ名
- スクリプトエディタ内のコード

ヒント : Windows では、JMP 環境設定でツールヒントを表示しないよう設定できます。［**ファイル**］>［**環境設定**］>［**一般**］を選択し、［**メニューのヒントを表示**］の選択を解除します。このオプションは、Macintosh では使用できません。

JMP User Community

JMP User Community では、さまざまな方法で JMP をさらに習得したり、他の SAS ユーザとのコミュニケーションを図ったりできます。ラーニングライブラリには 1 ページガイド、チュートリアル、デモなどが用意されており、JMP を使い始める上でとても便利です。また、JMP のさまざまなトレーニングコースに登録して、自己教育を進めることも可能です。

その他のリソースとして、ディスカッションフォーラム、サンプルデータやスクリプトファイルの交換、Webcast セミナー、ソーシャルネットワークグループなども利用できます。

Web サイトの JMP リソースにアクセスするには、［**ヘルプ**］>［**JMP User Community**］を選択するか、https://community.jmp.com/ をご覧ください。

JMPer Cable

JMPer Cable は、JMP ユーザを対象とした年刊の専門誌です。JMPer Cable は次の JMP Web サイトで閲覧可能です。

http://www.jmp.com/about/newsletters/jmpercable/（英語）

JMP 関連書籍

JMP 関連書籍は、次の JMP Web ページで紹介されています。

https://www.jmp.com/ja_jp/academic/books-for-jmp-users.html

「JMP スターター」ウィンドウ

JMP またはデータ分析にあまり慣れていないユーザは、「JMP スターター」ウィンドウから開始するとよいでしょう。カテゴリ分けされた項目には説明がついており、ボタンをクリックするだけで該当の機能を起動できます。「JMP スターター」ウィンドウには、［分析］、［グラフ］、［テーブル］、および［ファイル］メニュー内の多くのオプションがあります。また、JMP Pro の機能やプラットフォームのリストも含まれています。

- 「JMP スターター」ウィンドウを開くには、［**表示**］（Macintosh では［**ウィンドウ**］）**>**［**JMP スターター**］を選択します。
- Windows で JMP の起動時に自動的に「JMP スターター」を表示するには、［**ファイル**］**>**［**環境設定**］**>**［**一般**］を選び、「開始時の JMP ウィンドウ」リストから［**JMP スターター**］を選択します。Macintosh では、［**JMP**］**>**［**環境設定**］**>**［**起動時に JMP スターターウィンドウを表示する**］を選択します。

テクニカルサポート

JMP のテクニカルサポートは、JMP のエンジニアが担当し、その多くは、統計学などの技術的な分野の知識を有しています。

http://www.jmp.com/japan/support には、テクニカルサポートへの連絡方法などが記載されています。

第2章

JSL関数
関数、演算子、メッセージのまとめ

ここでは、JMPで用意されている多くの関数と演算子について、簡単に説明しています。また、一般的なオブジェクトメッセージについても簡単に説明しています。詳細は、JMPを起動した後、[**ヘルプ**] > [**スクリプトの索引**]で表示される「スクリプトの索引」を参照してください。

プラットフォームメッセージの詳細については、『スクリプトガイド』の 「プラットフォームのスクリプト」章 を参照してください。

割り当て関数

JSLには、**割り当て関数**も用意されています。割り当て関数は、演算結果を変数に直接、代入します。関数の形式で指定した場合、最初のオペランドに演算結果が割り当てられます。最も基本的な割り当て演算子は、等号が1つの=演算子です（対応する関数はAssign関数）。たとえば、*a*が3のとき、「a+=4」を実行すると、*a*が7になります。

割り当て関数の最初のオペランドは、値を割り当てることができる変数でなければなりません。このような変数は、「左辺値（**L-Value**）」と呼ばれています。たとえば、「3+=4」といった式は、「3」は単なる数値ですので、値を割り当てることはできません。そのため、この式はエラーとなります。しかし、「a+=4」といった式は、「*a*」が値を割り当てられる変数なので、実行できます。

Add To(a, b)

a+=b

説明

a と *b* を足して、その合計を *a* に代入する。

戻り値

列の合計

引数

a 変数でなければならない。

b 変数、数値、または行列。

ノート

第1引数の値は変更を受け入れる必要があるので、変数でなければなりません。第1引数を数値にすると、エラーが出ます。

Add To() という関数の形式で指定する場合、引数は2つしか指定できません。引数が1つまたはなしの場合、Add To() は欠測値を戻します。また、最初の2つの引数以外は無視されます。

a+=b という演算子の形式で指定する場合、3つ以上の引数を取ることができます。その場合、ペアを右から左へと評価し、それぞれの合計が左側の変数に代入されます。最後の引数を除いて、引数はすべて変数でなければなりません。

例

a+=b+=c

b と *c* を足して、その合計を *b* に代入します。さらに、*a* と *b* を足して、その合計を *a* に代入します。

次も参照

『スクリプトガイド』の「データ構造」章

Assign(a, b)

a=b

説明

b の値を *a* に代入する。

戻り値

a の新しい値

引数

a 変数でなければならない。

b 変数、数値、または行列。

ノート

a は値の変更を受け入れる必要があるので、変数でなければなりません。第 1 引数を数値にすると、エラーが出ます。*b* が何らかの式の場合、まずその式が評価され、その結果が *a* に代入されます。

Divide To(a, b)

a/=b

説明

a を *b* で割り、その結果を *a* に代入する。

戻り値

商

引数

a 変数でなければならない。

b 変数、数値、または行列。

次も参照

『スクリプトガイド』の「データ構造」章

Multiply To(a, b)

a*=b

説明

a と *b* を掛けて、その積を *a* に代入する。

戻り値

n の階乗

引数

a 変数でなければならない。

b 変数、数値、または行列。

ノート

第 1 引数の値は変更を受け入れる必要があるので、変数でなければなりません。第 1 引数を数値にすると、エラーが出ます。

Multiply To() という関数の形式で指定する場合、引数は 2 つしか指定できません。引数が 1 つまたはなしの場合、Multiply To() は欠測値を戻します。また、最初の 2 つの引数以外は無視されます。

a*=b という演算子の形式で指定する場合、3 つ以上の引数を取ることができます。その場合、ペアを右から左へと評価し、それぞれの合計が左側の変数に代入されます。最後の引数を除いて、引数はすべて変数でなければなりません。

例

a*=b*=c

b と *c* を掛けて、その積を *b* に代入します。さらに、*a* と *b* を掛けて、その積を *a* に代入します。

次も参照

『スクリプトガイド』の「データ構造」章

PostDecrement(a)

a--

説明

ポストデクリメント。*a* から 1 を引いて、差を *a* に代入する。

戻り値

a-1

引数

a 変数でなければならない。

ノート

a-- または Post Decrement(a) が別の式の中にある場合、まずその式が評価され、次にデクリメント演算子が実行されます。この式は、主にループ制御に使用されます。

PostIncrement(a)

a++

説明

ポストインクリメント。*a* に 1 を足して、合計を *a* に代入する。

戻り値

a+1

引数

a 変数でなければならない。

ノート

a++ または PostIncrement(a) が別の式の中にある場合、まずその式が評価され、次にインリメント演算子が実行されます。主にループ制御に使用されます。

Subtract To(a, b)

a-=b

説明

a から *b* を引いて、差を *a* に代入する。

戻り値

差

引数

a 変数でなければならない。

b 変数、数値、または行列。

ノート

第1引数の値は変更を受け入れる必要があるので、変数でなければなりません。第1引数を数値にすると、エラーが出ます。

Subtract To() という関数の形式で指定する場合、引数は2つしか指定できません。引数が1つまたはなしの場合、Subtract To() は欠測値を戻します。また、最初の2つの引数以外は無視されます。

a-=b という演算子の形式で指定する場合、3つ以上の引数を取ることができます。その場合、ペアを右から左へと評価し、それぞれの合計が左側の変数に代入されます。最後の引数を除いて、引数はすべて変数でなければなりません。

例

```
a-=b-=c
```

b から *c* を引いて、その差を *b* に代入します。さらに、*a* から *b* を引いて、その差を *a* に代入します。

次も参照

『スクリプトガイド』の「データ構造」章

文字関数

大部分の文字関数においては、引数や戻り値は文字列です（一部、数値であるものもあります）。引数に文字列定数を指定する場合、その文字列は引用符で囲む必要があります。

一部の関数については、『スクリプトガイド』の「データタイプ」章で詳しく解説しています。

Blob To Char(blob, <"encoding">)

説明

指定のエンコーディングを使って、BLOB から文字列を作成する。

戻り値

文字列

引数

blob BLOB（binary large object）

encoding（オプション）エンコーディングを指定する引用符付き文字列。文字列のデフォルトのエンコーディングは utf-8 です。utf-16le、utf-16be、us-ascii、iso-8859-1、ascii~hex、shift-jis、euc-jp もサポートされています。

ノート

エンコーディングのオプションのうち、*US-ASCII* は、CR、LF、TAB などが含まれる BLOB データを、特別な考慮をせずに、ASCII 文字の文字列に単純に変換します。

Blob To Matrix(blob, "type", bytes, "endian", <nCols>)

説明

blobのバイトを数値に変換して行列を作成する。

戻り値

BLOBを数値に変換した行列

引数

blob BLOBまたはBLOBへの参照。

type 数値のデータ型を示す引用符付き文字列。int、uint、floatのいずれか。

bytes BLOB内のデータのサイズをバイトで示したもの。1、2、4、8のいずれか。

endian エンディアンを表す引数。最初のバイトが最上位、最下位のいずれであるかを示す引用符付き文字列。指定できるパラメータは次のとおりです。

- "big"は、最初のバイトが最上位であることを示します（ビッグエンディアン）。
- "little"は、最初のバイトが最下位であることを示します（リトルエンディアン）。
- "native"は、コンピュータのネイティブ形式を使用します。

nCols（オプション）行列の列数を指定する。デフォルト値は1です。

Char(x, <width>, <decimal>)

説明

式または数値を文字列に変換する。

戻り値

文字列

引数

x 式または数値。式はExpr()で囲む必要があります。囲まない場合、その評価結果が文字列に変換されます。

width（オプション）文字列の最大文字数を設定する数値。

decimal（オプション）文字列に含まれる、小数点以下の最大桁数を設定する数値。

ノート

引数 *width* が引数 *decimal* より優先されます。

例

```
Char( Pi(), 10, 4)
  "3.1416"

Char( Pi(), 3, 4)
  "3.1"
```

Char To Blob(string, <"encoding">)

説明

文字列を、バイナリデータ（BLOB）に変換する。

戻り値

BLOB オブジェクト

引数

`string` 引用符付き文字列、または文字列変数。

`encoding`（オプション）エンコーディングを指定する引用符付き文字列。BLOB のデフォルトのエンコーディングは `utf-8` です。`utf-16le`、`utf-16be`、`us-ascii`、`iso-8859-1`、`ascii~hex`、`shift-jis`、`euc-jp` もサポートされています。

ノート

BLOB を印字可能な形式に変換する際、\（および ~ " ！と、ASCII の印字不能な範囲の文字）は、16 進数表記に変換されます（バックスラッシュの場合は、~5C）。

```
x = Char To Blob( "abc\def\!n" );
y = Blob To Char( x, encoding = "ASCII~HEX" );
If(
   y == "abc~5Cdef~0A", "JMP 12.2 以降のバージョンの動作 ",
   y == "abc\def~0A", "JMP 12.2 より前のバージョンの動作 "
);
  "JMP 12.2以降のバージョンの動作" // 出力
```

Char To Hex(value, <"integer" | "encoding">)

Hex(value, <"integer" | "encoding">)

説明

指定のエンコーディングを使って、値を 16 進数の文字列に変換する。

戻り値

16 進数文字列

引数

`value` 任意の数値、引用符付き文字列、または BLOB。

`integer` value が数値の場合、浮動小数点ではなく整数としての 16 進数を戻す。

`encoding`（オプション）エンコーディングを指定する引用符付き文字列。デフォルトのエンコーディングは `utf-8` です。`utf-16le`、`utf-16be`、`us-ascii`、`iso-8859-1`、`ascii~hex`、`shift-jis`、`euc-jp` もサポートされています。

Collapse Whitespace("text")

説明

先頭および末尾の空白文字を削除し、文字列内で空白文字が連続している部分は重複を削除する。つまり、`Collapse Whitespace` 関数で 2 つの連続したスペースを 1 つのスペースに置換できます。

戻り値

引用符付き文字列。

引数

`text` 引用符付き文字列。

Concat(a, b)

Concat(A, B)

a||b

A||B

説明

文字列の場合：文字列 *b* を文字列 *a* に追加する。どちらの引数も変更されません。

リストの場合：リスト *b* をリスト *a* に追加する。どちらの引数も変更されません。

行列の場合：行列 A と行列 B を横に連結する。

戻り値

文字列の場合：文字列 *a* の後に文字列 *b* を追加した文字列

リストの場合：リスト *a* の後にリスト *b* を追加したリスト

行列の場合：行列

引数

2 つ以上の文字列、文字列変数、リスト、または行列。

ノート

3 つ以上の引数を取ることができます。追加の文字列は、左から右の順番に文字列の最後に追加されます。行列の場合は、左から右の順番で、横に結合されていきます。

例

```
a = "こんにちは"; b = " "; c = "世界"; a || b || c;
  "こんにちは 世界"
d = {"りんご", "バナナ"}; e = {"桃", "梨"}; Concat( d, e );
  {"りんご", "バナナ", "桃", "梨"}
A = [1 2 3]; B = [4 5 6]; Concat( A, B );
  [1 2 3 4 5 6]
```

Concat Items

「Concat Items({string1, string2, ...}, <delimiter>})」（126ページ）を参照してください。

Concat To(a, b)

Concat To(a, b)

a||=b

A||=B

説明

文字列の場合：文字列 *a* に文字列 *b* を追加して、新しくできた連結文字列を *a* に代入する。

行列の場合：行列 *a* に行列 *b* を追加して、新しくできた行列を *a* に代入する。

リストの場合：リスト *a* にリスト *b* を追加して、新しくできたリストを *a* に代入する。

戻り値

文字列の場合：文字列 *a* の後に文字列 *b* を追加した文字列

行列の場合：行列

リストの場合：リスト *a* の後にリスト *b* を追加したリスト

引数

2つ以上の文字列、文字列変数、行列、またはリスト。第1引数には変数を指定しなければならない

ノート

3つ以上の引数を取ることができます。追加の文字列、行列、またはリストは、左から右の順番に文字列の最後に追加されます。

例

```
a = "こんにちは"; b = " "; c = "世界"; Concat To( a, b, c ); Show( a );
  a = "こんにちは 世界";
A = [1 2 3]; B = [4 5 6]; Concat To( A, B ); Show( A );
  A = [1 2 3 4 5 6];
d = {"りんご", "バナナ"}; e = {"桃", "梨"}; Concat to(d,e); Show( d );
  d = {"りんご", "バナナ", "桃", "梨"};
```

Contains(whole, part, <start>)

説明

部分（*part*）が全体（*whole*）の中に含まれているかどうかを判断する。

戻り値

part が見つかったとき、リスト、文字列、および名前空間の場合は *part* が最初に見つかった位置の数値を戻し、連想配列の場合は1を戻す。

part が見つからないときは、すべてのケースで0が戻されます。

引数

whole 文字列、リスト、名前空間、または連想配列。

part 文字列または名前空間の場合、文字列 *whole* の一部の文字列。リストの場合、リスト *whole* にある項目。連想配列の場合、マップ *whole* にあるキー。

start（オプション）文字列 whole 内での検索開始位置を表す数値引数。全体（*whole*）の中。なお、*start* が負の場合、Contains は、*whole* の長さから *start* を差し引いた位置から、*whole* 内の *part* を逆方向に検索します。連想配列の場合、*start* は意味がなく、無視されます。

例

```
nameList={"Katie", "Louise", "Jane", "Jaclyn"};
r = Contains(nameList, "Katie");
```

この例では、項目 "Katie" がリストの1番目にあるので、1が戻されます。

Contains Item(x, <item | list | pattern>, <delimiter>)

説明

多重応答を認識し、指定された項目、リスト、パターン、区切り文字を検索する。この関数は、多重応答の尺度または列プロパティを持つ列に対して使用できます。

戻り値

引数 *x* のテキスト内にある単語のいずれかと、単語（item）、単語リスト（list）の中の1つ、またはパターン（pattern）がマッチするかどうかを、ブール値で戻す。テキスト内の単語は、オプションで指定された区切り文字（delimiter）で区切られます。デフォルトの区切り文字はカンマ（","）です。なお、テキスト（*x*）から抽出された各単語の末尾にある空白は削除されます。

例

次の例では、**"pots"** およびそれに続くカンマを検索して、結果を出力します。

```
x = "Franklin Garden Supply is a leading online store featuring garden decor,
   statues, pots, shovels, benches, and much more.";
b = Contains Item( x, "pots", "," );
If( b,
   Write( "指定された項目が見つかりました。"  ), Write( "一致しません。" )
);
```

指定された項目が見つかりました。

Ends With(string, substring)

説明

文字列（string）の最後に部分文字列（substring）があるかどうかを判断する。

戻り値

文字列（string）が部分文字列（substring）で終わっていれば1、そうでなければ0

引数

string 引用符付き文字列、または文字列変数。リストでも可。
substring 引用符付き文字列、または文字列変数。リストでも可。

等価表現

```
Right(string, Length(substring)) == substring
```

Hex(value, <"integer" | encoding="enc")

「Char To Hex(value, <"integer" | "encoding">)」（31ページ）を参照してください。

Hex To Blob(string)

説明

16進数の文字列（*string*）（空白を含む）から、BLOB（binary large object）を作成する。

例

```
Hex To Blob( "4A4D50" );
```

Char To Blob("JMP", "ascii~hex")

Hex To Char(string, <encoding>)

説明

16進数の文字列（*string*）を、整数、もしくは、浮動小数点に変換します。

例

Hex To Char("30") の結果は “0” です。

ノート

文字列のデフォルトのエンコーディングは utf-8 です。utf-16le、utf-16be、us-ascii、iso-8859-1、ascii~hex、shift-jis、euc-jp もサポートされています。

Hex To Number(string)

説明

16 進数の文字列（*string*）を、整数、もしくは、浮動小数点に変換する。

例

```
Hex To Number( "80" );
  128
```

ノート

16 進数は、整数、もしくは、IEEE 754 の 64 ビット形式の浮動小数点数として、変換されます。

バイト間（つまり数字のペア）およびバイトの中央に空白文字が含まれていてもかまいません（例：FF 1919、F F1919）。

Insert

「Insert(source, item, <position>)」（127ページ）を参照してください。

Insert Into

「Insert Into(source, item, <position>)」（128ページ）を参照してください。

Item(n, string, <delimiters>)

説明

引数の文字列（*string*）から *n* 番目の単語を抽出する。区切り文字（*delimiters*）を指定すると、文字列を各単語に区切る際に、その文字が使われます。デフォルトの区切り文字はスペースです。複数の区切り文字を指定した場合、指定したすべての文字が区切り文字とみなされます。

ノート

Item() は、区切り文字 1 つ 1 つが個別の区切り文字として使われる以外は Word() と同じ。Word() では、連続した複数の区切り文字が 1 つの区切り文字として扱われます。

```
Item( 4,"the quick, brown fox", ", " );
  "brown"
Word( 4,"the quick, brown fox", ", " );
  "fox"
```

Left(string, n, <filler>)

Left(list, n, <filler>)

説明

元の文字列（*string*）から、右から n 文字の文字列を取り出す。もしくは、元のリスト（list）から、最初の *n* 個のリスト項目を取り出す。文字列（*string*）の長さが *n* 個よりも短い場合は、*filler* に指定された文字列が右側に補充されます。

Length(string)

説明

引用符付き文字列（*string*）の文字数（長さ）を計算する。

Lowercase(string)

説明

引用符付き文字列（*string*）内に現れる文字をすべて小文字に変換する。

Munger(string, offset, find|length)

Munger(string, offset, find, replace)

説明

引用符付き文字列（*string*）に文字を挿入したり、削除したりして、新しい文字列を作る。また、引数の指定方法により、文字列の一部を取り出す、また、ある文字列が何文字目に含まれているかを計算する、といった処理を実行できます。

offset は、string 内での検索開始位置を示す整数です。なお、ある検索文字列において、その先頭位置より *offset* が大きい場合、その検索文字列は検索に含まれません。また、*offset* が string の長さより大きい場合、*offset* は、（string の長さ +1）に設定されます。

Num(string)

説明

文字列を数値に変換する。

Regex("source", "pattern", (<replacementString>, <GLOBALREPLACE>), <format>, <IGNORECASE>)

説明

ソース文字列（*source*）内でパターン (*pattern*) を検索する。

戻り値

一致するテキストを文字列または数値で戻す。一致するテキストが見つからない場合は欠測値を戻します。

引数

source 引用符付き文字列。

pattern 引用符付き正規表現。

format（オプション）一致したグループへの前方参照。デフォルトは、一致した文字列全体である /0。/n は *n* 番目にマッチしたものを戻します。

IGNORECASE（オプション）IGNORECASE を指定しない場合、大文字と小文字が区別される。

GLOBALREPLACE 置換文字列と一緒に使用するオプション。一致したものがすべて見つかるまで、ソース文字列（source）に正規表現を適用します。

Remove

「Remove(source, position, <n>)」（128ページ）を参照してください。

Remove From

「Remove From(source, position, <n>)」（129ページ）を参照してください。

Repeat(source, a)

Repeat(matrix, a, b)

説明

source を、*a* 回、繰り返して連結した結果を戻す。または、matrix を *a* 回だけ縦に結合し、それを *b* 回だけ横に結合した行列を戻します。*source* はテキスト、またはリスト、matrix は行列です。

Reverse

「Reverse(source)」（129ページ）を参照してください。

Reverse Into

「Reverse Into(source)」（129ページ）を参照してください。

Right(string, n, <filler>)

Right(list, n, <filler>)

説明

元の文字列（*string*）から、右から n 文字の文字列を取り出す。もしくは、元のリスト（list）から、最後の *n* 個のリスト項目を取り出す。文字列（*string*）の長さが *n* 個よりも短い場合は、*filler* に指定された文字列が左側に補充されます。

Shift

「Shift(source, <n>)」（129ページ）を参照してください。

Shift Into

「Shift Into(source, <n>)」(130ページ)を参照してください。

Starts With(string, substring)

説明

文字列(string)の冒頭に部分文字列(*substring*)があるかどうかを判断する。

戻り値

文字列(*string*)が部分文字列(*substring*)で始まっていれば1、そうでなければ0

引数

string 引用符付き文字列または文字列変数。リストでも可。
substring 引用符付き文字列または文字列変数。リストでも可。

等価表現

```
Left(string, Length(substring)) == substring
```

Substitute

「Substitute(string, substring, replacementString, ...)」(130ページ)を参照してください。

Substitute Into

「Substitute Into(string, substring, replacementString, ...)」(131ページ)を参照してください。

Substr(string, start, length)

説明

第2引数(start)で指定した位置から第3引数(length)で指定した文字数の文字を、第1引数(string)の文字列から抽出する。第1引数には、文字列を指定してください。開始位置と文字数の引数は、整数です。

例

この例では、ファーストネームを抽出します。

```
Substr( "Katie Layman", 1, 5 );
```

1文字目から5文字を読み取り、残りの文字を無視して「Katie」を戻します。

Titlecase("text")

説明

文字列をタイトルケースに変換する。つまり、文字列内の各単語の先頭を大文字にし、残りを小文字にします。

戻り値

引用符付き文字列。

引数

text 引用符付き文字列。

例

次のスクリプトを実行すると、

```
Titlecase( "veronica layman ")
```

下記の文字列が戻されます。

```
"Veronica Layman"
```

Trim("text",<left|right|both>)

Trim Whitespace("text",<left|right|both>)

説明

最初および最後のスペースを削除する。

戻り値

引用符付き文字列。

引数

text 引用符付き文字列。

left|right|both 第2引数は、空白文字を文字列の左側（left）、右側（right）、または両側（both）のどちらから削除するかを指定します。第2引数が指定されない場合、両側から空白文字が削除されます。

例

次のコマンドを見てみましょう。

```
Trim( "  John    ", both )
```

下記の文字列が戻されます。

```
"John"
```

Uppercase(string)

説明

引用符付き文字列（*string*）内に現れる小文字をすべて大文字に変換する。

Word(n, "text", <"delimiters">)

説明

引数の文字列（text）から *n* 番目の単語を抽出する。区切り文字（delimiters）を指定すると、文字列を各単語に区切る際に、その文字が使われます。デフォルトの区切り文字はスペースです。複数の区切り文字を指定した場合、指定したすべての文字が区切り文字とみなされます。

例

この例では、ラストネームを戻します。

```
Word( 2, "Katie Layman" );
```

ノート

Word() 関数と Item() 関数は同じです。ただし、Item() 関数は、区切り文字が連続している時にそれらを 1 つ 1 つカウントしますが、Word() 関数は連続している区切り文字を 1 文字の区切り文字として扱います。

```
Item( 4,"the quick brown fox" );
  "fox"
Word( 4,"the  quick brown fox" ); // "quick" の前のスペースは 2 つ
  "brown"
```

Words

「Words("text", <"delimiters">)」（131 ページ）を参照してください。

XPath Query(xml, "xpath_expression")

説明

XML ドキュメントに対して、XPath 式を実行する。

戻り値

リスト

引数

xml 有効な XML ドキュメント。

xpath_expression 引用符付き XPath 1.0 式。

例

JMP で検定結果のレポートを作成し、重要な詳細を XML ドキュメントに書き出したとしましょう。検定の結果は <result> タグに挟まれています。

次の式は、その XML ドキュメントを変数内に保存します。XPath Query 式は、XML を解析して <result> に挟まれたテキストノードを見つけます。結果はリストで戻されます。

```
rpt =
"\[<?xml version="1.0" encoding="utf-8"?>
<JMP><report><title>Production Report</title>
<result>November 21st: Pass</result>
<result>November 22nd: Fail</result>
<note>Tests ran at 3:00 a.m.</note></report>
</JMP> ]\";
results = XPath Query( rpt, "//result/text()" );
  {"November 21st: Pass", "November 22nd: Fail"}
```

文字パターン関数

パターンマッチ表現の作成方法と使用方法の詳細については、『スクリプトガイド』の「データタイプ」章を参照してください。

Pat Abort()

説明

パターンマッチを即座に終了するパターンを生成する。バックアップも再試行も行いません。条件の割り当ては行われません。すでに行われた即座の割り当てが維持されます。

戻り値

マッチが終了したときは 0

引数

なし

Pat Altern(pattern1, <pattern 2, ...>)

説明

パターン引数のいずれかにマッチするパターンを生成する。

戻り値

パターン

引数

1 つ以上のパターン。

Pat Any("string")

説明

引数のいずれか 1 文字にマッチするパターンを生成する。

戻り値

パターン

引数

string 文字列

Pat Arb()

説明

任意の文字列にマッチするパターンを生成する。始めはヌル文字列とマッチします。バックアップが行われるたびにマッチする文字列が 1 文字ずつ長くなります。

戻り値

パターン

引数

なし

例

```
p = "最後ではなく " + Pat Arb() >?stuffInTheMiddle + "出てきます";
Pat Match( "ストーリーの最後ではなく始めの方で、3 匹のクマが出てきます", p );
Show( stuffInTheMiddle );
```

stuffInTheMiddle = "始めの方で、3匹のクマが"

Pat Arb No(pattern)

説明

引数（*pattern*）に0回以上マッチするパターンを生成する。

戻り値

パターン

引数

pattern マッチ対象のパターン。

例

```
adjectives = "Lサイズ" | "Mサイズ" | "Sサイズ" | "温かい" | "冷たい" | "熱い" | "甘め";
rc = Pat Match( "Mサイズの甘めの熱い紅茶を1つください",
                Pat Arbno( adjectives | Pat Any("の") ) >> adj +
                ("紅茶" | "コーヒー" | "ミルク") );
Show( rc, adj );
  rc = 1;
  adj = "Mサイズの甘めの熱い";
```

Pat At(varName)

説明

ヌル文字列にマッチするパターンを生成し、ソース文字列の現在の場所を、指定したJSL変数（*varName*）に代入する。割り当ては即座に行われます。値が割り当てられた変数をexpr()で使用することにより、残りのマッチを変更することができます。

戻り値

パターン

引数

varName 結果を格納する変数の名前。

例

```
p = ":" + Pat At( listStart ) + Expr(
   If( listStart == 1,
      Pat Immediate( Pat Len( 3 ), early ),
      Pat Immediate( Pat Len( 2 ), late )
   )
);
early = "";
late = "";
Pat Match( ":123456789", p );
Show( early, late );
early = "";
late = "";
Pat Match( "   :123456789", p );
Show( early, late );
```

まず次が生成されます。

```
early = "123"
late = ""
```

その後、次が生成されます。

```
early = ""
late = "12"
```

Pat Break("string")

説明

引数に含まれていない0個以上の文字にマッチするパターンを生成し、引数に含まれる文字の前でパターンマッチを停止させる。引数に含まれる文字が見つからない場合は失敗します（特に、停止のための文字が見つからないままソース文字列を最後まで検索し終わった場合）。

戻り値

パターン

引数

string 文字列

Pat Concat(pattern1, pattern2 <pattern 3, ...>)

Pattern1 + Pattern2 + ...

説明

引数で指定したパターンに順番にマッチするパターンを生成する。

戻り値

パターン

引数

2つ以上のパターン。

Pat Conditional(pattern, varName)

説明

マッチが終了し成功した場合、パターンマッチの結果を第2引数の変数（*varName*）に保存する。

戻り値

パターン

引数

pattern マッチ対象のパターン。

varName 結果を格納する変数の名前。

例

```
type = "未定義";
rc = Pat Match(
   "青りんご",
   Pat Conditional( "赤" | "青", type ) + "りんご"
);
```

```
Show( rc, type );
  rc = 1;
  type = "青";
```

Pat Fail()

説明

マッチングを先に進ませないパターンを生成する。マッチングはバックアップし、別のマッチングを試行します。別のマッチングが残っていない場合、マッチングは失敗し、Pat Match が 0 を戻します。

戻り値

マッチが失敗したときは 0

引数

なし

Pat Fence()

説明

マッチングが先に進む場合は、成功し、ヌル文字列にマッチするパターンを生成する。バックアップが生じた場合は、失敗します。一部のマッチの最適化に使用できます。

戻り値

マッチが成功したときは 1、そうでなければ 0

引数

なし

Pat Immediate(pattern, varName)

説明

パターンマッチの結果を第 2 引数（*varName*）の名前の変数に即座に保存する。

戻り値

パターン

引数

pattern マッチ対象のパターン。
varName 結果を格納する変数の名前。

例

```
type = "未定義";
rc = Pat Match(
   "青りんご",
   ("赤" | "青") >> type + "なし"
);
Show( rc, type );
  rc = 0
  type = "青"
```

マッチが失敗した場合でも、割り当ては即座に行われます。

Pat Len(int)

説明

n 文字にマッチするパターンを生成する。

戻り値

パターン

引数

int 文字数を指定する整数。

Pat Match(SourceText, Pattern, <ReplacementText>, <NULL>, <ANCHOR>, <MATCHCASE>, <FULLSCAN>)

説明

Pat Match は、ソーステキスト（*SourceText*）に対してパターン（*Pattern*）を実行する。直接指定する方法か、または別の JSL 変数へパターンを割り当てておく方法により、パターンを生成する必要があります。

戻り値

パターンが見つかれば 1、そうでなければ 0

引数

SourceText 検索対象のテキストを示す文字列または文字列変数。

Pattern 検索対象のテキストを示すパターンまたはパターン変数。

ReplacementText（オプション）ソーステキスト内でマッチしたパターンを指定した文字列に置換する。

NULL ANCHOR、MATCHCASE、または FULLSCAN が必要で、かつ置換テキストがない場合の第 3 引数のプレースホルダー。

ANCHOR（オプション）文字列の冒頭でパターンマッチを開始するためのオプション。次のパターンマッチは文字列の冒頭にパターン「cream」がないため、失敗となります。

```
Pat Match( "coffee with cream and sugar", "cream", NULL, ANCHOR );
```

MATCHCASE（オプション）パターンマッチで大文字／小文字を区別するためのオプション。デフォルトの Pat Match() は大文字／小文字を区別しません。

FULLSCAN（オプション）Pat Match にすべての候補を強制的に検索させるオプション。検索数が多いほど、多くのメモリが必要となります。デフォルトでは、Pat Match() は FULLSCAN を使用せず、再帰を停止する、あるいはマッチングを継続するための前提を決めて動作します。

Pat Not Any("string")

説明

引数内に含まれていない 1 つの文字にマッチするパターンを生成する。

戻り値

パターン

引数

string 文字列

Pat Pos(int)

説明

現在の位置が文字列の左端から *int* のとき、ヌル文字列にマッチするパターンを生成する。その他の場合は失敗します。

戻り値

パターン

引数

`int` 文字列内での位置を指定する整数

Pat R Pos(int)

説明

現在の位置が文字列の右端から *int* のとき、ヌル文字列にマッチするパターンを生成する。その他の場合は失敗します。

戻り値

パターン

引数

`int` 文字列内での位置を指定する整数

Pat R Tab(int)

説明

文字列の最後から int の位置までにマッチするパターンを生成する。0 文字以上の文字とマッチできます。後ろに移動する場合、または文字列の最後を超える場合は失敗します。

戻り値

パターン

引数

`int` 文字列内での位置を指定する整数

Pat Regex(string)

説明

指定された正規表現とマッチするパターンを生成する。正規表現を記述する引数 *string* は、引用符付きの文字列で指定する必要があります。

戻り値

パターン

引数

`string` 文字列

Pat Rem()

説明

文字列の残りにマッチするパターンを生成する。Pat R Tab(0) と等価です。

戻り値

パターン

引数

なし

Pat Repeat(pattern, minimum, maximum, GREEDY|RELUCTANT)

説明

指定されたパターン（*pattern*）に最小（*minimum*）～最大（*maximum*）回マッチするパターンを生成する。

戻り値

パターン

引数

pattern マッチ対象のパターン。

minimum 最大回数。最小回数より小さい整数。

maximum 最小回数。最大回数より大きい整数。

GREEDY|RELUCTANT GREEDY が指定されている場合、まず最大回数だけ試行してから最小回数まで戻る。RELUCTANT が指定されている場合、まず最小回数だけ試行してから最大回数まで進む。

ノート

Pat Arbno(p) は Pat Repeat(p,0,infinity,RELUCTANT) と同じです。

Pat Repeat(p) は Pat Repeat(p,1,infinity,GREEDY) と同じです。

Pat Repeat(p,n) は Pat Repeat(p,n,infinity,GREEDY) と同じです。

Pat Repeat(p,n,m) は Pat Repeat(p,n,m,GREEDY) と同じです。

Pat Span("string")

説明

引数内の文字で構成されている 1 文字以上（0 ではなく）の文字列にマッチするパターンを生成する。これは GREEDY のため、常に、可能な限り長い文字列とのマッチを行います。マッチする文字が 0 文字の場合は失敗します。

戻り値

パターン

引数

string 文字列

Pat String("string")

説明
文字列引数にマッチするパターンを生成する。

戻り値
パターン

引数
string 文字列

Pat Succeed()

説明
バックアップが行われた場合でも常に成功するパターンを生成する。ヌル文字列とマッチします。

戻り値
マッチが成功したときは 1

引数
なし

Pat Tab(int)

説明
ソース文字列の *int* の位置までマッチするパターンを生成する。0 文字以上の文字とマッチできます。後ろに移動する場合、または文字列の最後を超える場合は失敗します。

戻り値
パターン

引数
int 文字列内での位置を指定する整数

Pat Test(expr)

説明
式（*expr*）が 0 以外のときは、成功し、ヌル文字列とマッチするパターンを生成する。そうでない場合は失敗します。

戻り値
パターン

引数
expr 式。

ノート
通常、引数には expr() が使用されます。通常、テストは、Pat Immediate、Pat Conditional、および Pat At から戻された現在の値に対して行います。expr() を使用しないと、テストは、パターンマッチを行っている最中ではなく、パターンを生成した時点の値によって行われます。この値により、パター

ンマッチは常に成功するか常に失敗するかのどちらかとなってしまうため、多くの場合、意図したとおりに動作しません。

例

```
nCats = 0;
whichCat = 3;
string = "生米生麦生卵";
rc = Pat Match(
   string,
   "生" + Pat Test(
      Expr(
         nCats = nCats + 1;
         nCats == whichCat;
      )
   ),
   "ゆで"
);
Show( rc, string, nCats );
  rc = 1
  string = "生米生麦ゆで卵"
  nCats = 3
```

Regex Match(source, pattern, <MATCHCASE>)

説明

引用符付き文字列で指定されたソース（*source*）に対して、パターン（*pattern*）のマッチを実行する。

戻り値

パターン

引数

source 文字列
pattern パターン
MATCHCASE（オプション）MATCHCASE を指定しない場合、大文字と小文字は区別されない。

コメント関数

// comment

説明

行の終わりまでコメント。

ノート

スクリプトの実行時、// から後はすべて無視されます。

/* comment */

説明

行の途中にも挿入できるコメント。

ノート

スクリプトの実行時、開始タグ /* と終了タグ */ の間のものはすべて無視されます。この形式のコメントは、引数のリスト内も含め、ほぼどこにでも挿入できます。ただし、引用符付き文字列の中にコメントを挿入した場合、そのコメントは文字列の一部として扱われ、コメントとはみなされません。また、コメントを演算子の真ん中に挿入することはできません。

例

```
+/*comment*/=
:/*comment*/name
```

これは無効で、エラーが出ます。最初のコメントは += を妨害し、2 番目のコメントは :name を妨害しています。

```
sums = {(a+b /*comment*/), /*comment*/ (c^2)}
```

これは有効な JSL です。どちらのコメントも無視されます。

//!

説明

これをスクリプトの 1 行目に記述すると、そのスクリプトを JMP で開いたときに、スクリプトエディタに開かれずに、すぐにスクリプトが実行される。

ノート

このコマンドが指定されていても、ファイルを実行しないで、開くこともできます。［**ファイル**］>［**開く**］を選択して、呼び出されたダイアログにて、JSL ファイルを選択し、Ctrl キーを押しながら［**開く**］をクリックします。または、ホームウィンドウの「最近使ったファイル」に表示されているファイルを右クリックし、［**スクリプトの編集**］を選択します。いずれかの操作により、スクリプトは実行されずに、スクリプトウィンドウに表示されます。

/*debug step*/

/*debug run*/

説明

これをスクリプトの 1 行目に記述した場合、そのスクリプトは、実行時にデバッガで開かれる。

ノート

小文字のみ使用できます。debug と step の間、または debug と run の間にスペースを 1 つ挿入し、それ以外にはスペースを挿入しません。どちらか一方の行を、スクリプトの 1 行目に記述します。1 行目が空白で、2 行目にこのコメントを記述した場合、デバッグコマンドは有効になりません。

比較関数

比較演算子（<、<=、>、>=）は、数値、文字列、および行列に対して使用できます。行列の場合、各要素を比較した結果の行列が生成されます。文字列と数値や文字列と行列など、タイプが異なるオペランドを比較すると、結果は欠測値になります。リストを比較することはできず、その場合も欠測値が戻されます。

等価演算子（==と!=）は、数値、文字列、行列、およびリストに対して使用できます。行列の場合、各要素が等しいかどうかを表す結果の行列が生成されます。リストの場合は、リスト全体として等しいかどうかを表す結果の値が1つ生成されます。文字列と数値や文字列と行列など、タイプが異なるオペランドが**等しいかどうかをテスト**すると、0（等しくない）という結果になります。

範囲をチェックする演算子を使用すると、2つの指定した値の範囲に入っているかどうかをチェックできます。

```
a = 1;
Show( 1 <= a < 3 );
b = 2;
Show( 2 < b <= 3 );
  1 <= a < 3 = 1;
  2 < b <= 3 = 0;
```

比較演算子が含まれた式は、順にではなく、すべて一度に評価される

比較演算子はすべて**省略演算子**（複数の演算を指定する際に省略ができる演算子）です。大部分の演算においては、一度に1つずつ演算子が評価されていきます。しかし、比較演算子で結合されているオペランドは1つの大きなまとまりとして取り扱われます。1つのまとまりとして評価すると、部分ごとに評価する通常の方法とは異なる結果が生成されます。たとえば、次の2つのステートメントはそれぞれ別のものです。

```
12 < a < 13;
(12 < a) < 13;
```

最初のステートメントは、3つの引数と両方の演算子すべてが読み取られて一緒に評価されるため、*a*が12と13の範囲にあるかどうかをチェックします。2つ目のステートメントでは、括弧を使って明示的に処理をグループに分けて、左から順に評価します。そのため、この式では、まず12が*a*より小さいかどうかをチェックし、真のとき1を、偽のとき0を戻します。次に、その結果（0または1）が13より小さいかどうかをチェックします。0も1も13より小さいので、この結果は必ず真になります。

同じ演算子の組み合わせまたは異なる演算子の組み合わせ（<... <=と <=... <）で使用すると、すべての比較演算子は一度に評価されます。つまり、一度に1つずつ比較演算子を評価する場合には、括弧( )を使って、明示的に操作の順序を制御する必要があることを意味します。

Equal(a, b, ...)

a==b==...

説明

指定されたすべての値を比較し、同じ値かどうかを判定する。

戻り値

すべての引数の結果が同じ値になるときは1（true）

そうでなければ0（false）

引数

2つ以上の変数、参照、行列、または数値。

ノート

3つ以上の引数が指定された場合は、すべての引数が同じ値である場合にだけ1が戻されます。この演算子は、主に、条件文や、ループの制御で使用されます。

文字列の比較では、大文字と小文字が区別されます。

Greater(a, b, ...)

a>b>...

説明

指定された値を比較して、各ペアについて左の値が右の値よりも大きいかどうかを判定する。

戻り値

a が *b* より大きい（および *b* が *c* より大きい）ときは1（true）

そうでなければ0（false）

引数

2つ以上の変数、参照、行列、または数値。

ノート

3つ以上の引数が指定された場合は、すべての引数の値がそれぞれの右側の引数の値よりも大きい場合にだけ1が戻されます。この演算子は、主に、条件文や、ループの制御で使用されます。

Greater、Less、GreaterOrEqual、および LessOrEqual を同時に指定することもきます。括弧を使ってグループ化した場合を除き、各ペアは左から右へと評価されます。括弧を使って明確に式の評価順を示すこともできます。

Greater or Equal(a, b, ...)

a>=b>=...

説明

指定された値を比較して、各ペアについて左の値が右の値以上かどうかを判定する。

戻り値

a が *b* 以上（および *b* が *c* 以上）のときは1（true）

そうでなければ0（false）

引数

2つ以上の変数、参照、行列、または数値。

ノート

3つ以上の引数が指定された場合は、すべての引数の値がそれぞれの右側の引数の値以上である場合にだけ1が戻されます。この演算子は、主に、条件文や、ループの制御で使用されます。

Greater、Less、GreaterOrEqual、およびLessOrEqualを同時に指定することもきます。括弧を使ってグループ化した場合を除き、各ペアは左から右へと評価されます。括弧を使って明確に式の評価順を示すこともできます。

Is Missing(expr)

説明

評価後の引数が欠測値のときは1、そうでなければ0を戻す。

Less(a, b, ...)

a<b<...

説明

指定された値を比較して、各ペアについて左の値が右の値よりも小さいかどうかを判定する。

戻り値

*a*が*b*より小さい（および*b*が*c*より小さい）ときは1（true）

そうでなければ0（false）

引数

2つ以上の変数、参照、行列、または数値。

ノート

3つ以上の引数が指定された場合は、第1引数が第2引数以下で、そのほかのすべての引数の値がそれぞれの右側の引数未満である場合にだけ1が戻されます。この演算子は、主に、条件文や、ループの制御で使用されます。

Greater、Less、GreaterOrEqual、およびLessOrEqualを同時に指定することもきます。括弧を使ってグループ化した場合を除き、各ペアは左から右へと評価されます。括弧を使って明確に式の評価順を示すこともできます。

Less LessEqual(a, b, c, ...)

a<b<=c<=...

説明

範囲の照合。bがaより大きくc以下であるかを判定する。

戻り値

*b*が*a*より大きく*c*以下のときは1（true）

そうでなければ0（false）

引数

a, b, c 変数、参照、行列、または数値

ノート

第1引数が第2引数より小さく、残りの各引数が右側の引数以下であるという、2つの条件が両方とも満たされた場合に1を返します。この演算子は、主に、条件文や、ループの制御で使用されます。

Less or Equal(a, b, ...)

a<=b<=...

説明

指定された値を比較して、各ペアについて左の値が右の値以下であるかどうかを判定する。

戻り値

a が *b* 以下（および *b* が *c* 以下）のときは1（true）

そうでなければ0（false）

引数

2つ以上の変数、参照、行列、または数値。

ノート

3つ以上の引数が指定された場合は、すべての引数の値がそれぞれの右側の引数の値以下である場合にだけ1が戻されます。この演算子は、主に、条件文や、ループの制御で使用されます。

Greater、Less、GreaterOrEqual、および LessOrEqual を同時に指定することもきます。括弧を使ってグループ化した場合を除き、各ペアは左から右へと評価されます。括弧を使って明確に式の評価順を示すこともできます。

LessEqual Less(a, b, c, ...)

a<=b<c<...

説明

範囲の照合。b が a 以上で c より小さいかを判定する。

戻り値

b が *a* 以上で *c* より小さいときは1（true）

そうでなければ0（false）

引数

a，b，c 変数、参照、行列、または数値

ノート

第1引数が第2引数以下で、残りの各引数が右側の引数より小さいという、2つの条件が両方とも満たされた場合に1を返します。この演算子は、主に、条件文や、ループの制御で使用されます。

Not Equal(a, b)

a!=b

説明

a と b を比較して、等しくないかどうか判定する。

戻り値

a と *b* が同値と評価されたときは 0（false）

そうでなければ 1（true）

引数

a, b 任意の変数または数値。

ノート

主に、条件文や、ループの制御で使用されます。

条件付き関数と論理関数

And(a, b)

a&b

説明

論理積。

戻り値

a と *b* の両方が真のときは 1（true）

a または *b* のどちらか、または *a* と *b* の両方が偽のときは 0（false）

a または *b* のどちらか、または *a* と *b* の両方が欠測値のときは欠測値

引数

2 つ以上の変数または式。

ノート

3 つ以上の引数を取ることができます。a&b は、すべての引数が真と評価されたときのみ、1（true）を戻します。

AndMZ(a, b)

AndV3(a, b)

a:&b

説明

JMP3 の動作を持った論理 And で、欠測値を 0 として処理する。

戻り値

a と *b* の両方が真のときは 1（true）

a または *b* のどちらか、または *a* と *b* の両方が偽のときは 0（false）

a または *b* のどちらか、または *a* と *b* の両方が欠測値のときは 0（false）

引数

2 つ以上の変数または式。

ノート

3つ以上の引数を取ることができます。a:&b は、すべての引数が真と評価された場合のみ、1（true）を戻します。JMP 3 のデータテーブルを開くと、この関数がすべての And 関数に自動的に使用されます。

Break()

説明

ループを完全に停止して、ループの後に続くステートメントの実行に移る。

ノート

Break は、For ループ、While ループ、および For Each Row で使用できます。

Choose(expr, r1, r2, r3, ..., rElse)

説明

式（*expr*）を評価し、*expr* の値が 1 のときは *r1* の値を戻し、2 のときは *r2* の値を戻す。どれにも該当しない場合は最後の引数（*rElse*）を戻します。

戻り値

引数リスト内でインデックスが *expr* とマッチする値、または最後の引数の値。

引数

expr 式または値。

r1, r2, r3, ... 式または値。

Continue()

説明

ループの現在の反復を終了して、次の反復を開始する。

ノート

Continue は、For ループ、While ループ、および For Each Row で使用できます。

For(init, while, increment, body)

説明

body に指定されたスクリプトを、条件（*while*）が真である間、繰り返し実行する。*init* と *increment* によって反復を制御します。

戻り値

なし

引数

init ループ制御カウンタの初期化。

while ループの継続／終了の条件。条件文の while が真である限り、ループは再度反復される。while が偽になると、即座にループは終了する。

increment ループが実行されるたびに、while が評価された後、ループカウンタをインクリメント（またはデクリメント）する

body 任意の数の有効な JSL 式。複数ある場合は接合する

例

```
mysum = 0; myprod = 1;
For( i = 1, i <= 10, i++, mysum += i; myprod *= i; );
Show( mysum, myprod );
  mysum = 55;
  myprod = 3628800;
```

For Each Row(<dt,> script)

説明

データテーブルの各行に対してスクリプト（*script*）を繰り返す。

戻り値

なし

引数

dt（オプション）位置指定引数。データテーブルへの参照。この引数が割り当ての形式をとらない場合は、データテーブルの式とみなされます。

script 任意の有効な JSL 式

例

次の例は、データテーブルへの参照を設定し、「Big Class.jmp」のすべての行に処理を適用します。ある行の「**年齢**」の値が 15 より大きい場合は、その年齢がログに出力されます。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
For Each Row( dt, If( :年齢 > 15, Show( :年齢 ) ) );
```

If(condition, result, condition, ..., <elseResult>)

説明

条件（*condition*）が真のとき、結果（*result*）を戻す。そうでない場合は、次の条件（*condition*）が真になったときの結果（*result*）を戻します。それより前の条件がどれも真でない場合に、オプションの *elseResult* を戻します。*elseResult* が指定されておらず、どの条件も真ではない場合は、何も起こりません。

IfMax(expr1, result1, expr2, result2, ... <all missing result>)

説明

引数の各ペアにおいて、1 つ目の引数が最大値となっているペアの結果（各ペアの 2 つ目）を戻します。最大値となる式が複数ある場合には、最初の最大値に対応する結果を戻します。すべての式が欠測値で、最終結果が指定されていない場合、欠測値を戻します。すべての式が欠測値であり、最後の結果式が指定されている場合は、最後の結果式の評価を戻します。条件式は数値になる必要がありますが、結果式はどのような値でもかまいません。

戻り値

最大値に対する結果式の評価

IfMin(expr1, result1, expr2, result2, ... <all missing result>)

説明

引数の各ペアにおいて、1つ目の引数が最小値となっているペアの結果（各ペアの2つ目）を戻します。最小値となる式が複数ある場合には、最初の最小値に対応する結果を戻します。すべての式が欠測値で、最終結果が指定されていない場合、欠測値を戻します。すべての式が欠測値であり、最後の結果式が指定されている場合は、最後の結果式の評価を戻します。条件式は数値になる必要がありますが、結果式はどのような値でもかまいません。

戻り値

最小値に対する結果式の評価

IfMZ(condition, result, condition, ..., <elseResult>)

IfV3(condition, result, condition, ..., <elseResult>)

説明

JMP バージョン 3.x の論理 `If` で、欠測値は 0 として扱われる。バージョン 3.x のデータテーブルを開くときに自動的に使われます。

Interpolate(x, x1, y1, x2, y2)

Interpolate(x, xmatrix, ymatrix)

説明

（*x1, y1*）と（*x2, y2*）、または行列 *xmatrix* と *ymatrix* で指定された 2 点間を線形に補間し、与えられた *x* 値に対応する *y* 値を戻す。各点は、昇順で並んでいる必要があります。

Is Associative Array(name)

説明

評価後の引数が連想配列のときは 1、そうでなければ 0 を戻す。

Is Empty(global)

Is Empty(dt)

Is Empty(col)

説明

global 変数、データテーブル、またはデータ列に値がない（初期化されていない）ときは 1、あれば 0 を戻す。

Is Expr(x)

説明

評価後の引数が式のときは 1、そうでなければ 0 を戻す。

Is List

「Is List(x)」（128ページ）を参照してください。

Is Name(x)

説明

評価後の引数が名前のときは1、そうでなければ0を戻す。

Is Namespace(namespace)

説明

引数 namespace が名前空間の場合は1、そうでなければ0を戻す。

Is Number(x)

説明

評価後の引数が数値か欠測値のときは1、そうでなければ0を戻す。

Is Scriptable(x)

説明

評価後の引数がスクリプト可能なオブジェクトのときは1、そうでなければ0を戻す。

Is String(x)

説明

評価後の引数が文字列のときは1、そうでなければ0を戻す。

Match(a, value1, result1, value2, result2, ...)

説明

a が値1（*value1*）に等しいときは結果1（*result1*）を戻し、*a* が値2（*value2*）に等しいときは結果2（*result2*）、・・・を戻す。

MatchMZ(a, value1, result1, value2, result2, ...)

MatchV3(a, value1, result1, value2, result2, ...)

説明

JMP バージョン3.x の Match に相当し、欠測値は0として扱われる。バージョン3.x のデータテーブルを開くときに自動的に実行されます。

Not(a)

!a

説明

論理否定。

戻り値

a が 0 以外のときは 0（false）

a=0 のときは 1（true）

a が欠測値のときは欠測値

引数

a 任意の変数または数値。変数の場合、数値または行列でなければなりません。

ノート

主に、条件文や、ループの制御で使用されます。

Or(a, b)

a|b

説明

論理和。

戻り値

a と *b* のどちらかまたは両方が真のときは 1（true）

そうでなければ 0（false）

どちらかが欠測値でもう一方が偽の場合、および、両方とも欠測値の場合は 0（false）

引数

a, b 任意の変数または数値。

ノート

主に、条件文や、ループの制御で使用されます。

OrMZ(a, b)

OrV3(a, b)

a :| b

説明

JMP3 の動作を持った論理和 Or で、欠測値を 0 として処理する。

戻り値

a と *b* のどちらかまたは両方が真のときは 1（true）

そうでなければ 0（false）

引数

a, b 任意の変数または数値。

ノート

主に、条件文や、ループの制御で使用されます。JMP 3 のデータテーブルを開くと、この関数がすべての Or 関数に自動的に使用されます。

Or() は、いずれかの引数の評価が欠測値の場合、欠測値を戻します。OrMZ() は、いずれかの引数の評価が欠測値の場合、0 を戻します。

```
Step(x0, x1, y1, x2, y2, ...)
Step(x0, [x1, x2, ...], [y1, y2, ...])
```

説明

x0 以下となっている引数 *x* のうち、最大の値に対応する引数 *y* を戻す。引数 *x* は小さい順に指定されていなければなりません。

```
Stop()
```

説明

実行中のスクリプトをそこですぐに停止する。

```
While(expr, body)
```

説明

条件式（*expr*）を評価し、真であれば本文の式（*body*）を実行する。これを条件式（*expr*）が真でなくなるまで繰り返します。

```
Zero Or Missing(expr)
```

説明

式（*expr*）が欠測値またはゼロを生成する場合に 1 を戻し、そうでない場合は 0 を戻す。

定数関数

JMP には、便利な定数の関数が 2 つあります。

メモ: これらの関数は、引数は不要ですが、括弧は必要です。

```
e()
```

説明

定数 *e* の値（2.7182818284590451...）を戻す。

Pi( )

説明

定数π（3.1415926535897931...）を戻す。

日付と時間関数

JMPでは、日付時間値を、1904年1月1日午前0時からの秒数で内部的に保持します。

「x=01Jan1904」という式を実行すると、指定されている日付がJMPでの基準日つまり「ゼロ日」のため、*x*はゼロに設定されます。通常、日付値の内部的な数値は、かなり大きな数値になります。たとえば、5oct1998は2990390400になります。

Abbrev Date(date)

説明

指定された日付（*date*）を文字列に変換する。

戻り値

日付の文字列

引数

date 基準日（1904/01/01の午前零時）からの秒数、または任意の日付時間演算子

例

```
Abbrev Date( 29Feb2004 );
```

2004/02/29

次も参照

『スクリプトガイド』の「データタイプ」章

As Date(x)

説明

数値または式*x*の日付型の形式を戻す。戻り値は、テキストウィンドウに送られたときに、日付または期間として表示されるようになります。1年以上を示す値は日付として戻されます。1年未満を示す値は期間として戻されます。

戻り値

指定された数値または式から計算された日付

引数

x 数値または式。

次も参照

『スクリプトガイド』の「データタイプ」章

Date Difference(datetime1, datetime2, "interval_name", <"alignment">)

説明

2つの日付時間値の差を、指定された単位で戻す。

戻り値

数値

引数

datetime1、datetime2 日付時間値。

interval_name 期間を示す引用符付き文字列。"Month"、"Day"、"Hour" など。

alignment（オプション）文字列。指定できるパラメータは次のとおりです。

- "start" は、2つの日付時間値の間隔内で、指定した期間が何度始まったかを数えます。
- "actual" は、2つの日付時間値の間隔内に期間全体がいくつ含まれていたかを数えます。
- "fractional" は、2つの日付時間値の間隔内の期間の数を、"Year"、"Quarter"、"Month" の長さの平均値を使って小数部まで計算します。

Date DMY(day, month, year)

説明

引数から日付値を生成する。

戻り値

1904年1月1日午前0時から指定の日付までの秒数

引数

day 数値、日付、1 ～ 31。エラーのチェックは行われません。たとえば、2月31日も入力されてしまうので注意が必要

month 数値、月、1 ～ 12

year 数値、年

Date Increment(datetime, "interval_name", <increment>, <"alignment">)

説明

開始の日付時間値に、指定された期間を加算する。

戻り値

新しい日付時間値

引数

datetime 開始の日付時間値。

interval_name 期間を示す引用符付き文字列。"Year"、"Quarter"、"Month"、"Week"、"Day"、"Hour"、"Minute"、"Second" がサポートされています。

increment（オプション）間隔数を指定する数値。デフォルト値は1です。

alignment（オプション）文字列。指定できるパラメータは次のとおりです。

- "start" は、指定した日付時間値から interval で指定した単位より小さな部分を切り捨てます。たとえば、時間を切り捨てて、日付を出力します。"start" がデフォルト値です。

– "actual"は、指定した日付時間値を切り捨てることなく加算を行います。
– "fractional"は、日付時間値の間隔数を、"Year"、"Quarter"、"Month"の長さの平均値を使って小数部まで計算します。

Date MDY(month, day, year)

説明
引数から日付値を生成する。

戻り値
1904年1月1日午前0時から指定の日付までの秒数

引数
month 数値、月、1～12
day 数値、日付、1～31。エラーのチェックは行われません。たとえば、2月31日も入力されてしまうので注意が必要
year 数値、年

Day(datetime)

説明
日付時間値（*datetime*）の日付の値を戻す。

戻り値
指定した日付（date）がその月の何日であるかを表す整数値を戻す。

引数
datetime 1904年1月1日午前0時から指定の日付までの秒数。または式でも可

例

```
d1 = Date DMY( 12, 2, 2003 );
  3127852800

Day( 3127852800 );
  12
Day( d1 );
  12
```

Day Of Week(datetime)

説明
日付時間値（*datetime*）の曜日の値を戻す。

戻り値
指定の日付（date）が何曜日であるかを表す整数値を戻す。

引数
datetime 1904年1月1日午前0時から指定の日付までの秒数。または式でも可

Day Of Year(datetime)

説明

日付時間値（*datetime*）の日付が、1 年のうちで何日目かを戻す。

戻り値

指定された日付が、その年の何日目であるかを表す整数値を戻す。

引数

datetime 1904 年 1 月 1 日午前 0 時から指定の日付までの秒数。または式でも可

Format(x, "format", <"currency code">, <decimal>)

Format Date(x, "format", <"currency code">, <decimal>)

説明

x の値を、第 2 引数で指定した形式（"*format*"）に変換する。秒数を日付時間値で表示するときに使われます。選択できる形式は、データテーブルの列プロパティに表示されます。

戻り値

指定の形式に変換した日付

引数

x 列、数値、または日付時間値

format 任意の有効な形式（"Best"、"Fixed Decimal"、"Percent"、"PValue"、"Scientific"、"Currency"、またはいずれかの Date/Time 形式）を引用符付き文字列で指定する。

currency code（オプション）特定の通貨を示す ISO 4217 コード（たとえば、英国ポンドの場合は "GBP"）。この引数は、Currency が形式（*format*）に指定されている場合にのみ有効です。

decimal（オプション）表示する小数桁数を整数で指定する。

Hour(datetime, <12|24>)

説明

日付時間値（*datetime*）の時間の値を戻す。

戻り値

指定された日付時間値（datetime）が、24 時間もしくは 12 時間のうちで何時間目になっているかを、整数で戻す

引数

datetime 1904 年 1 月 1 日午前 0 時から指定の日付までの秒数。または式でも可

12|24 形式を、12 時間形式にする（午前および午後における時間）。デフォルト値は 24 時間形式です。

HP Time()

説明

高精度の時間値（マイクロ秒単位）を戻す。この関数は、別の HP Time () 値との比較で役立ちます。時間値は、JMP セッションの開始から経過したマイクロ秒数を表します。

ノート

これより精度の低い時間値については、Tick Seconds() を使用します。

In Days(n)

説明

日数 *n* を秒数に変換して戻す。秒数を In Days(1) で割ると、日数に変換できます。

In Format(string, "format")

Parse Date(string, "format")

説明

指定された形式（*"format"*）で文字列（*string*）を解析し、日付／時間の値を戻す。As Date() と同様に、日付を "ddMonyyyy" 形式で戻します。選択できる形式は、データテーブルの列プロパティに表示されます。As Date の詳細については、「ユーティリティ関数」（240 ページ）を参照してください。

例

```
Informat( "07152000", "MMDDYYYY" );
```

15Jul2000

In Hours(n)

説明

時間数 *n* を秒数に変換して戻す。秒数を In Hours(1) で割ると、時間数に変換できます。

In Minutes(n)

説明

分数 *n* を秒数に変換して戻す。秒数を In Minutes(1) で割ると、分数に変換できます。

In Weeks(n)

説明

週数 *n* を秒数に変換して戻す。秒数を In Weeks(1) で割ると、週数に変換できます。

In Years(n)

説明

年数 *n* を秒数に変換して戻す。秒数を In Years(1) で割ると、年数に変換できます。

Long Date(date)

説明

与えられた日付（*date*）から、"2004 年 02 月 29 日 日曜日 " や "2011 年 11 月 9 日 水曜日 " のような、OS で指定されているロケールの日付表示を戻す。

MDYHMS(date)

説明

与えられた日付（*date*）から "2/29/04 00:02:20" のような形式の日付表示を戻す。

Minute(datetime)

説明

日付時間値（*datetime*）の分の値（0 ～ 59）を戻す。

戻り値

指定した日付時間値（datetime）の分を表す整数値

Month(date)

説明

指定した日付（*date*）の月を数値で戻す。

Parse Date()

「In Format(string, "format")」（66ページ）を参照してください。

Quarter(datetime)

説明

日付時間値（*datetime*）の四半期の値（1 ～ 4）を戻す。

Second(datetime)

説明

日付時間値（*datetime*）の秒の値を戻す。

戻り値

指定した日付時間値（datetime）の秒を表す整数値

引数

datetime 1904 年 1 月 1 日午前 0 時から指定の日付までの秒数。または式でも可

Short Date(date)

説明

与えられた日付（*date*）を mm/dd/yy の形式で戻す。たとえば、2004 年 2 月 29 日は "02/29/2004" となります。

Tick Seconds()

説明

スクリプトの実行にかかった時間を 60 分の 1 秒まで計測する。

ノート

これより高精度の時間値については（マイクロ秒など）、HP Time() 関数を使用します。

Time Of Day(datetime)

説明

指定された日時（*datetime*）がその日の何秒目かを整数値で戻す。

Today()

説明

1904 年 1 月 1 日午前 0 時から現在の日時までの秒数を戻す。引数はとりませんが、括弧は必要です。

Week Of Year(date, <rule_*n*>)

説明

指定された日時（date）がその年の何週目であるかを戻す。1 年の第 1 週がいつ始まるかを決めるルール（rule）は 3 つあります。

- ルール 1（rule_1、デフォルト）では、1 週間は日曜日に始まり、1 年の初めの日曜日は第 2 週の始まりとみなされます。そのため、第1週に該当する日数は6日以下になり、0の場合もあります。
- ルール2（rule_2）では、第1日曜日が第1週の始まりとなります。日曜日より前の曜日は第0週とみなされます。
- ルール3（rule_3）では、ISO-8601準拠の週番号が返されます。1週間は月曜日から始まります。1年の第1週は、1月4日を含む週とされます。1年の最初または最後の3日間は直近の年の週番号となる場合があります。

Year(date)

説明

与えられた日付（*date*）の年を数値で戻す。

離散型確率関数

Beta Binomial Distribution(k, p, n, delta)

説明

ベータ二項分布の累積分布関数。ベータ二項分布に従う確率変数が *k* 以下になる確率を戻す。

Beta Binomial Probability(k, p, n, delta)

説明

ベータ二項分布の確率関数。ベータ二項分布に従う確率変数が *k* と等しくなる確率を戻す。

Beta Binomial Quantile(p, n, delta, cumprob)

説明

ベータ二項分布の分位点関数。ベータ二項分布（p, n, delta）の累積確率が *cumprob* 以上となるような整数の分位点のうち、最小のものを戻す。

Binomial Distribution(p, n, k)

説明

二項分布の累積分布関数。二項分布に従う確率変数が *k* 以下になる確率。

戻り値

各試行の成功確率が *p*、試行回数が *n*、成功回数が *k* である二項分布の累積分布関数の値

引数

p 各試行の成功確率
n 試行回数
k 成功回数

Binomial Probability(p, n, k)

説明

二項分布の確率関数。二項分布に従う確率変数が *k* と等しくなる確率。

戻り値

成功確率が *p* のときに、*n* 回の試行のうち *k* 回成功する確率

引数

p 各試行の成功確率
n 試行回数
k 成功回数

Binomial Quantile(p, n, cumprob)

説明

二項分布の分位点関数。

戻り値

ベータ二項分布（p, n）の累積確率が *cumprob* 以上となるような整数の分位点のうち、最小のものを戻す。

引数

p 各試行の成功確率
n 試行回数

cumprob 累積確率

Gamma Poisson Distribution(k, lambda, sigma)

説明

ガンマ Poisson 分布の累積分布関数。ガンマ Poisson 分布に従う確率変数が *k* 以下になる確率を戻す。

引数

k 対象となる度数
lambda 平均の引数。
sigma 過分散の引数。

Gamma Poisson Probability(k, lambda, sigma)

説明

ガンマ Poisson 分布の確率関数。ガンマ Poisson 分布に従う確率変数が、*k* に等しくなる確率を戻す。

引数

k 対象となる度数
lambda 平均の引数。
sigma 過分散の引数。

Gamma Poisson Quantile(lambda, sigma, cumprob)

説明

ガンマ Poisson 分布の分位点関数。ガンマ Poisson 分布（lambda, sigma）の累積確率が *cumprob* 以上となるような整数の分位点のうち、最小のものを戻す。

Hypergeometric Distribution(N, K, n, x, <r>)

説明

超幾何分布の *x* における累積分布関数の値を戻す。母集団のサイズは *N*、母集団のなかで対象となるカテゴリに属する個数は *K*、標本サイズは *n*、標本のなかで対象カテゴリに属する個数は *x*、オプションのオッズ比は *r* で指定されます。

Hypergeometric Probability(N, k, n, x, <r>)

説明

超幾何分布に従う確率変数が *x* と等しくなる確率を戻す。

引数

N 母集団のサイズ
k 母集団の中で対象となるカテゴリに属する個数
n 標本サイズ
x 対象となる度数
r（オプション）オッズ比

Neg Binomial Distribution(p, n, k)

説明

負の二項分布の確率関数。成功確率が *p* のときに、*n* 回成功するまでに失敗する回数が、*k* 回以下である確率を戻す。

Neg Binomial Probability(p, n, k)

説明

負の二項分布の確率関数。成功確率が *p* のときに、*n* 回成功するまでに *k* 回、失敗する確率を戻す。

Poisson Distribution(lambda, k)

説明

指定の平均（*lambda*）を持つ Poisson 分布の、*k* における累積分布関数の値を戻す。

Poisson Probability(lambda, k)

説明

指定の平均（*lambda*）を持つ Poisson 分布に従う確率変数が、*k* に等しくなる確率を戻す。

Poisson Quantile(lambda, cumprob)

説明

Poisson 分布の分位点関数。Poisson 分布（lambda, sigma）の累積確率が *cumprob* 以上となるような整数の分位点のうち、最小のものを戻す。

表示関数

Alpha Shape(Triangulation)

説明

指定された三角分割に対して、そこから計算されるアルファシェイプを戻す。

Border Box(<Left(pix)>, <Right(pix)>, <Top(Pix)>, <Bottom(Pix)>, <Sides(0)>, db)

説明

別のディスプレイボックスを含むことができる枠付きのディスプレイボックスを作成する。外側の枠線と、その中身との間隔を、Left（左）、Right（右）、Top（上）、Bottom（下）というオプションで指定できます。Sides オプションによって、周りに描く枠、枠の色（黒もしくは強調色）、背景（透明もしくは白）、背景を消すかどうかを指定できます。

戻り値

ディスプレイボックス

引数

Left ピクセル数を指定する整数
Right ピクセル数を指定する整数
Top ピクセル数を指定する整数
Bottom ピクセル数を指定する整数
Sides ディスプレイボックスの設定に関する整数
db ディスプレイボックスオブジェクト（たとえば、テキストボックスや別のボーダーボックスなど）。

ノート

Sides オプションには、1*top + 2*left + 4*bottom + 8*right + 16*highlightcolor + 32*whitebackground + 64*erase という計算式で求められた整数を指定してください。たとえば、上（top）と下（bottom）に、黒色の枠線（hightlightcolor）を描きたい場合は、1+4 = 5 を指定してください。同じものを、白の背景色（whitebackground）に変えて、描きたい場合は、5+32 = 37 を指定してください。

Button Box("title", <<Set Icon("path"), "script" <<Set Icon Location("left, "right")

説明

クリックするとスクリプト（*script*）を実行する、*title* という名前のボタンを作成する。

戻り値

ディスプレイボックス（ボタンボックス）

引数

title 引用符付き文字列、または文字列変数。
script JSL スクリプト
<<Set Icon("path") パス名にあるイメージをボタン上に表示する。GIF、JPG、PNG、BMP、TIF などの一般的なグラフィック形式に対応しています。このオプションを指定しなかった場合、テキストだけのボタンが作成されます。title オプションの引数に空の文字列を指定し、アイコンだけを指定した場合、アイコンだけが表示されたボタンが作成されます。両方のオプションを指定すると、テキストとアイコンの両方が表示されたボタンが作成されます。その場合、アイコンは、テキストの隣に表示されます。
<<Set Icon Location("right" または "left") ボタン上のアイコンをテキストの左または右に配置する。

ノート

ボタンボックス内の改行文字は無視されます。

Calendar Box("title", < <<Date, <<Min Date, <<Max Date, <<Show Time>)

説明

日時を選択できる、ポップアップカレンダーを作成する。

戻り値

ディスプレイボックス（カレンダーボックス）

引数

title 引用符付き文字列、または文字列変数。
<<Date 現在選択されている日付。
<<Min Date 選択可能な最初の日付。
<<Max Date 選択可能な最後の日付。
<<Show Time 時間の選択を可能にする。

例

次の例では、カレンダーを作成し、1989 年 10 月 5 日が選択されている状態で表示します。また、選択可能な最初の日付と最後の日付を指定し、ユーザが選択可能な期間を限定します。

```
New Window( "カレンダーボックスの例", cal = Calendar Box() );
date = Date MDY (10, 5, 1989);
cal << Date( date );
cal << Show Time( 0 ); // 時間を使用しない

cal << Min Date( Date Increment(date, "Day", -60, "start" ) );
// 選択可能な最初の日付は、1989 年 10 月 5 日の 60 日前
// "start" を指定して、時間を切り捨てる

cal << Max Date( Date Increment(date, "Day",  60, "start" ) );
// 選択可能な最後の日付は、1989 年 10 月 5 日の 60 日後

cal << Set Function( Function( {this}, Print( Abbrev Date(this << Get Date()) ) )
  );
// 短い形式の日付をログに出力する
```

Check Box(list, <script>)

説明

1 つまたは複数のチェックボックスを表示するディスプレイボックスを作成する。

戻り値

ディスプレイボックス（チェックボックス）

引数

リスト 文字列を含むリスト
script（オプション）JSL スクリプト

メッセージ

<<Get(n) *n* で指定されたチェックボックス項目が選択されている場合は 1、そうでない場合は 0 を戻す。
<<Set(n, 0|1) *n* で指定されたチェックボックス項目を選択（1）または選択解除（0）する。
<<Get Selected 選択されているチェックボックス項目の名前を含む文字列リストを戻す。
<<Enable Item(n, 0|1) *n* で指定されたチェックボックス項目を有効（1）または無効（0）にする。無効になっているチェックボックスの状態は変更できません。
<<Item Enabled(check box item) 指定されたチェックボックス項目が有効の場合は 1、そうでない場合は 0 を戻す。

例

「one」、「two」、「three」というラベルがついた3つのチェックボックスを作成します。最初のチェックボックスが選択されます。

```
New Window( "例", Check Box( {"one", "two", "three"}, <<Set( 1, 1 ) ) );
```

Col List Box(<"all"|"character"|"numeric">, <width(pixels)>, <maxSelected(n)>, <nlines(n)>, <MaxItems(n)>, <MinItems(n)>, <On Change(expr)>,<script>)

説明

データテーブルの列を選択できるリストボックスを表示するディスプレイボックスを作成する。

戻り値

ディスプレイボックス（列リストボックス）

引数

`all | character | numeric`（オプション）現在のデータテーブルのすべての列をリストに追加するコマンド。"all" を省略すると、“ オプション ” ラベルの付いた空の列リストボックスが表示されます。“ オプション (文字)” を表示する場合は "character" を指定します。“ オプション (数値)” を表示する場合は "numeric" を指定します。

`width(pixels)`（オプション）リストボックスの幅を `pixels` に設定するコマンド。`pixels` はピクセル数を示す数値。

`maxSelected(n)`（オプション）選択できる項目が 1 つだけかどうかを設定するコマンド。*n*<1 の場合、*n* は無視される。

`nlines(n)`（オプション）リストボックスの長さを n 行に設定するコマンド。*n* は整数

`script`（オプション）スクリプト

`MaxItems(n)`（オプション）リストに追加できる最大列数

`MinItems(n)`（オプション）リストに必要な最小列数。たとえば、*n*=2 の場合、`Col List Box` の最初の 2 つの項目に、「必須」、「必須 (文字)」、もしくは「必須 (数値)」と表示される。

`On Change(expression)`（オプション）列リストボックスの内容が変更される度に式を評価するコマンド。この引数を持つ 2 つの列リストボックスの間をドラッグすると、両方の式が評価されます。ドラッグ先の式がまず評価され、その後ドラッグ元の式が評価されます。

メッセージ

`<<Set Tips ( {"Tip text 1", "Tip text 2", ...} )` リストボックス内の項目にツールヒントを設定する。ヌル文字列または空のリストを指定した場合、ツールヒントは設定されません。リストボックス内の項目のリストよりも、指定したリストの方が短い場合、最後のツールヒントテキストが、リストボックス内の残りのリスト項目にも使用されます。

`<<Set Tip ( "Tip text" ) Set Tips()` 関数で設定されたツールヒントを上書きする。ボックス自体にヒントが設定されている場合、個々のリスト項目のヒントを設定することはできません。

`Set Tip()` を引数未指定で使用すると、リストボックスのヒントが消去され、各リスト項目のツールヒントが表示されるようになります。

ノート

`maxSelected` の引数は、項目を 1 つだけ選択できるか、または複数選択できるかだけを決定するものです。2 つ以上の列を強制的に選択させるものではありません。

Column Dialog(ColList("rolename"), specifications)

説明

列の役割を指定するダイアログボックスを作成する。

戻り値

送られたコマンドおよびクリックされたボタンのリスト

引数

ColList ボタンと列の選択リストを作成。最低1つは指定。MaxCol（最大選択列数）、DataType（データタイプ）などを引数に指定可。

specifications ダイアログボックス上に追加する項目

Combo Box(list, <script>)

説明

メニュー付きのコンボボックスを表示するディスプレイボックスを作成する。

戻り値

ディスプレイボックス（コンボボックス）

引数

リスト 文字列を含むリスト

script（オプション）JSL スクリプト

Context Box(displayBox, ...)

説明

変数や式のスコープを限定するディスプレイボックスを定義する。各コンテキストボックスは、独立して、個別に評価・実行されます。

戻り値

ディスプレイボックス

引数

ディスプレイボックスの数

Current Journal()

説明

現在のジャーナルの最上位のディスプレイボックスへの参照を戻す。

戻り値

現在のジャーナルの最上位のディスプレイボックスへの参照

Current Report()

説明

現在のレポートウィンドウの最上位のディスプレイボックスへの参照を戻す。

戻り値

現在のレポートウィンドウの最上位のディスプレイボックスへの参照

Current Window()

説明

現在のウィンドウへの参照を戻す。

Dialog(contents)

Dialog は廃止されました。Dialog 関数の代わりに、Modal 引数とともに New Window() 関数を使用してください。詳細については、『スクリプトガイド』の「表示ツリー」章を参照してください。

Excerpt Box(report, subscripts)

説明

レポートから一部分を抜粋し、その抜粋したディスプレイボックスを戻す。レポートの番号を *report* で指定し、ディスプレイのリストを *subscripts* で指定します。subscripts に指定する番号は、抜粋したディスプレイを除いた後の番号。

Expr As Picture(expr(...), <width(pixels)>)

説明

計算式エディタに表示される式と同じ形式で、expr() をイメージに変換する。

戻り値

イメージへの参照

引数

expr(...) イメージとして表示できる有効な JSL 式を expr() の中に挿入

width(pixels)（オプション）ボックスの幅を pix に設定するコマンド。pix はピクセル数を示す数値。

Filter Col Selector(<data table(name)>, <width(pixels)>, <Nlines(n)>, <script>, <OnChange(expr)>)

説明

列名のリストを含むディスプレイボックスを戻す。列のフィルタリングを設定できる。

Global Box(global)

説明

グローバル変数（*global*）の値を直接編集するボックスを作成する。

Graph()

「Graph Box(properties, script)」（77ページ）を参照してください。

Graph 3D Box(properties)

説明

3次元散布図を含むディスプレイボックスを作成する。

戻り値

ディスプレイボックス

引数

プロパティには、framesize(x, y)、Xname("title")、Yname("title")、Zname("title")がある。

ノート

この機能は試験段階です。

Graph Box(properties, script)

Graph(properties, script)

説明

X軸およびY軸のあるグラフを作成する。

戻り値

ディスプレイボックス（グラフボックス）

引数

properties 引数には、title("title")、XScale(low, high)、YScale(low, high)、FrameSize(h, v)、XName("x")、YName("y")、SuppressAxesがある。

script グラフボックスで実行するスクリプト

H Center Box(display box)

説明

引数 display box に含まれているすべての子ディスプレイボックスを水平方向に中央揃えで配置して戻す。

H List Box(<Align("center"|"bottom")>, display box, <arguments>)

説明

ディスプレイボックスを作成し、その中に別のディスプレイボックスを横に並べて表示する。

引数

Align ディスプレイボックスの位置を、下揃え（bottom）または中央揃え（center）に指定する。デフォルトでは、下揃えで配置されます。

display box リストボックスに含める任意の数のディスプレイボックスを指定する。

H Sheet Box(<<Hold(report), display boxes)

説明

引数に指定された複数のディスプレイボックスを横方向に配置したディスプレイボックスを戻す。<<Hold() メッセージは、抜粋元となるレポートをどのシートが保持するかを示す。

H Splitter Box(<size(*h*,*v*)>, display box, <arguments>)

説明

引数に指定された複数のディスプレイボックスを横方向（パネル）に配置し、分割線で仕切ったディスプレイボックスを戻す。ユーザは分割線をドラッグしてパネルのサイズを変更できます。

引数

display box 分割線ボックスには任意の数のディスプレイボックスの引数を設定できる。

オプションの引数

size(*h*,*v*) Splitter Box のサイズをピクセルで指定する。このサイズは外側の Splitter Box 用です。内側のディスプレイボックスのサイズは、外側の Splitter Box の幅と高さに合わせて変更されます。

<<Size(*n*) 最後のパネルの比率を指定する。<<Size(.25) は、最後のパネルのサイズを分割線ボックスの高さ（H Splitter Box の場合は幅）の 25% に変更します。

<<Set Sizes({*n*,*n*}) 各パネルの比率を指定する。db<<Set Sizes({.75, .25}) は、最初のパネルのサイズを分割線ボックスの高さ（H Splitter Box の場合は幅）の 75% に変更し、2 番目のパネルのサイズを同 25% に変更します。

<<Close Panel(*n*, <*Boolean*>) 指定したパネルを閉じる。<<Close Panel(2) は、2 番目のパネルを閉じます。3 つ以上のパネルがある場合は、2 番目のブール値を含める必要があります。その値により、閉じたパネルによって余ったスペースを埋めるパネルを指定します。

- <<Close Panel(2,0) は、2 番目のパネルを閉じ、その後のパネルが余ったスペースを埋めます。
- <<Close Panel(2,1) は、2 番目のパネルを閉じ、その前のパネルが余ったスペースを埋めます。

<<Open Panel(*n*, <*Boolean*>) 指定したパネルを開く。3 つ以上のパネルがある場合は、2 番目のブール値を含める必要があります。前述の <<Close Panel と同様に機能します。パネルは初期状態ですべて開いているため、<<Open Panel は <<Close Panel を使用した後にのみ使用します。

<<Get Sizes() 各パネルの比率をリストで戻す。

Hier Box("text", Hier Box(...), ...)

説明

テキスト（text）が含まれているツリーのノードを作成する。これは、特性要因図プラットフォームの出力と同じです。Hier Box は別の Hier Box を含めることができ、ツリーを作成できます。テキストは Text Edit Box でもかまいません。

Icon Box("name")

説明

アイコンを含むディスプレイボックスを作成する。引数で指定する名前（name）には、Popup 、Locked 、Labeled 、Sub 、Excluded 、Hidden 、Continuous 、Nominal 、Ordinal があります。引数には、イメージへのパスを指定することもできます。

引数

`name` JMP のアイコン名またはアイコンのパスを示す引用符付き文字列

例

`Icon Box( "Nominal" )` は、名義尺度アイコンを含むディスプレイボックスを作成します。

`Icon Box( "$SAMPLE_IMAGES/pi.gif" )` は、pi.gif サンプルイメージを挿入します。

If Box(Boolean, display boxes)

説明

条件によって中身が表示されるディスプレイボックスを作成する。

引数

ブール値。1 の場合は `If Box` 内のディスプレイボックスを表示し、0 の場合は表示しません。

`display boxes` 任意のディスプレイボックスツリー

Journal Box("Journal Text")

説明

引用符付き文字列（*journal text*) から生成されるジャーナルを表示するディスプレイボックスを作成する。手動でジャーナルテキストを生成することはお勧めしません。

Line Seg(x, y, <Row States(dt | dt, [rows] | dt, {{rows}, ...} | {states}) >, <Sizes(s)>)

説明

つながった線分を描くディスプレイセグメントを戻す。オプションの第 3 引数では、データテーブルから行属性を割り当てるか、または独自に行属性を割り当てることができます。

Lineup Box(<NCol(n)>, <Spacing(pixels, <vspace>), display boxes, ...)

説明

n 列にボックスを整列させて表示するディスプレイボックスを作成する。

List Box({"item", ...}, <width(pixels)>, <maxSelected(n)>, <nLines(n)>, <script>)

説明

リストボックスを表示するディスプレイボックスを作成する。引数 item には、項目名の文字列を含むリストを指定します。もしくは、項目名の文字列と、尺度または並べ替え順序を示す文字列を含むリストのリストを指定します （項目名の大文字と小文字は、デフォルトで区別されます）。後者の場合、リストボックス内の項目の横に、尺度または並べ替え順序のアイコンが表示されます。

Marker Seg(x, y, <Row States(dt | dt, [rows] | dt, {{rows}, ...}| {states}) >, <Sizes(s)>)

説明

指定された x と y のすべての値にマーカーを描くディスプレイセグメントを作成する。オプションの第 3 引数では、データテーブルの行属性を使用するか、またはそれとは別のものを使用するかを指定できます。

Matrix Box(matrix)

説明

与えられた行列（*matrix*）を表示する。

Mouse Box(displayBoxArgs, messages)

説明

ドラッグ& ドロップ、マーキング、クリック、トラックなどのマウス動作に対する JSL コールバックを作成するボックスを戻す。

引数

displayBoxArgs Text Box や Button Box など、ユーザが操作できるオブジェクトを指定する。詳細については、［ヘルプ］メニューの［スクリプトの索引］を参照してください。

New Image()

New Image(width, height)

New Image("filepath")

New Image(picture)

New Image(matrix)

New Image("rgb"|"r"|"rgba", {matrix, ...})

説明

レポート内に新しいイメージを作成する。PNG、JPG、GIF、BMP、TIF のファイル形式がサポートされています。

戻り値

イメージ

引数

すべての引数セットはオプションだが、それぞれのセット内ではすべての引数が必須。

`width, height` イメージの幅と高さをピクセルで設定する。

`"filepath"` イメージへのファイルパス

`picture` JSL ピクチャーオブジェクト

`matrix` イメージを JSL カラーのピクセルの配列で示したもの

`"rgb"|"r"|"rgba", {matrix, ...}` チャネルを指定し（rgb、r、または rgba）、各チャネルの値の行列（0.0-1.0）を指定する。例：

```
New Image( "r", [r matrix] );
New Image( "rgb", {[r matrix], [g matrix], [b matrix]} );
New Image( "rgba", {[r matrix], [g matrix], [b matrix], [a matrix]} );
```

New Window("title", <arguments>, displayBox)

説明

指定されたタイトル（title：この引数は必須）をつけた新しいウィンドウと、ディスプレイボックスツリーを作成する。

その他の引数

`<<Script(<"script">)` スクリプトウィンドウを作成する。オプションで、スクリプトウィンドウに表示するスクリプトを、引用符付き文字列 *"script"* で指定できます。

`<<Journal` 空のジャーナルを作成する。

`<<Modal` 新しいウィンドウをモーダルウィンドウにする。このウィンドウを閉じるまで JMP 内で他のアクションを実行することができなくなります。［**OK**］ボタンまたは［**キャンセル**］ボタンを追加しなかった場合、それらは自動的に追加されます。**メモ：**この引数を使用する場合は、ウィンドウタイトルのすぐ後に続く第 2 引数として指定します。使用できるモーダルウィンドウの引数は次のとおりです。

- `<<On Open(expr)` は、ウィンドウの作成時に式（`expr`）を実行する。

> **メモ：**データテーブルでは、別のプログラムを実行する `On Open`（または `OnOpen`）スクリプトは、実行されません。このスクリプトが実行されるタイミングを制御する必要がある場合は、環境設定の［OnOpen スクリプトの評価］を設定してください。

- `<<On Close(expr)` は、ウィンドウが閉じるときに式（`expr`）を実行する。ウィンドウを閉じることができない場合は 0 を返します。
- `<<On Validate (expr)` は、［**OK**］ボタンが押されたときに式（`expr`）を実行する。True を返した場合はウィンドウが閉じ、それ以外の場合、ウィンドウは開いたままになります。
- `<<Return Result` は、ウィンドウが閉じるときの戻り値を、廃止された `Dialog()` 関数の戻り値と一致するよう変更する。

Show Toolbars(0|1) ツールバーの表示／非表示を切り替える。デフォルト値は 1 です。(Windows のみ)。

Show Menu(0|1) メニューバーの表示／非表示を切り替える。デフォルト値は 1 です。(Windows のみ)。

Suppress AutoHide メニューおよびツールバーの自動非表示機能の使用／不使用を切り替える。デフォルト値は 1 です。(Windows のみ)。

ノート

Dialog() は、JMP 10 で廃止されました。Dialog 関数の代わりに、Modal 引数とともに New Window() 関数を使用してください。New Window() の使用法の詳細については、『スクリプトガイド』の「表示ツリー」章を参照してください。

Number Col Box("title", numbers)

説明

指定されたタイトル（*title*）をつけた列を作成し、リストまたは行列の形で与えられた数値を挿入する。

Number Col Edit Box("title", numbers)

説明

指定されたタイトル（*title*）をつけた列を作成し、リストまたは行列の形で与えられた数値を挿入する。この関数によって作成された列の数値は、編集できます。

Number Edit Box(initValue, <width>)

説明

initValue 引数で指定された数値が入力された数値ボックスを作成する。

戻り値

ディスプレイボックスオブジェクト

引数

initValue 初期値として使用する数値。日付または時間の形式を使用すると、日付または時間のセレクタウィンドウが作成されます。

<width> ボックスの幅を文字数で設定する。

Outline Box("title", display box, ...)

説明

指定された複数のディスプレイボックスを含んだ、*title* という名前の新しいアウトラインを作成する。

Page Break Box()

説明

ウィンドウが印刷された場合、改ページを強制するディスプレイボックスを作成する。

Panel Box("title", display box)

説明

指定されたタイトル（*title*）のラベルをもつパネルのディスプレイボックスを作成する。そのパネルには、複数のディスプレイボックスを含めることができます。

Picture Box(Open(picture), format)

説明

グラフィックピクチャーオブジェクトを含むディスプレイボックスを作成する。

戻り値

ディスプレイボックスへの参照

引数

Open ピクチャを含むディレクトリを開く。

picture 含めるピクチャーのパス名。

format グラフィックファイルの形式。JMP でピクチャーを開く形式を指定します。この引数を指定しないと、ピクチャーはデフォルトのグラフィックプログラムで開かれます。

例

```
New Window( "例",
   Picture Box( Open( "$SAMPLE_IMAGES/pi.gif", gif ) ) );
```

Plot Col Box("title", numbers)

説明

数字（*numbers*）をグラフ化したディスプレイボックスに、引用符付き文字列で指定されたタイトル（*title*）を付けて戻す。数字にはリストまたは行列を指定できます。

Popup Box({"command1", script1, "command2", script2, ...})

説明

赤い三角形ボタンのメニューを作成する。引数のリストには、コマンドの文字列と実行されるスクリプトをペアで指定します。まず、コマンド文字列を指定し、その後にそのコマンドを選択したときに評価される式を指定します。コマンドが空の場合は、区切り線が挿入されます。**メモ：**ALT キーを押しながら赤い三角ボタンのメニューを右クリックすると、コマンドのチェックボックスがあるウィンドウが表示されます。詳細については、『JMP の使用法』の「JMP のレポート」章を参照してください。

Radio Box({"item", ...}, <script>)

説明

一連のラジオボタンを表示するディスプレイボックスを作成する。ラジオボタンが選択されるたびに、オプションのスクリプトが実行されます。

Range Slider Box(minValue, maxValue, lowVariable, highVariable, script)

説明

Range Slider Box() は、最小値（minValue）から最大値（maxValue）までの範囲スライダを表示したディスプレイボックスを戻す。2 つのスライダの位置が変化すると、その値が lowVariable と highVariable に代入され、スクリプトが実行されます。

戻り値

ディスプレイボックス（範囲スライダボックス）

引数

minValue, maxValue スライダの最小値および最大値となる数値
lowVariable 最小値側のスライダによって設定・変更される変数
highVariable 最大値側のスライダによって設定・変更される変数
script スライダボックスの移動に伴って実行される任意の有効な JSL コマンド

Report(obj)

説明

プラットフォームオブジェクト（*obj*）の表示ツリーを戻す。同じ処理は、次のように、メッセージとしてプラットフォームに送っても行えます。

```
obj<<Report
```

Scene Box(x size, y size)

説明

3D グラフィック用のシーンボックスを作成する。ボックスの横幅は *x size*、高さは *y size* に設定される。

Script Box(<"script">, <width>, <height>)

説明

引用符付き文字列で指定されたスクリプト（*script*）を含む編集ボックスを作成する。このボックスはスクリプトウィンドウの一種で、JSL を編集・実行できます。

引数

script（オプション）引用符付き文字列。これがスクリプトボックスに表示される
width（オプション）スクリプトボックスの幅を指定する整数
height（オプション）スクリプトボックスの高さを指定する整数

Scroll Box(<size(*h*,*v*)>, <flexible(Boolean)>, display box, ...)

説明

スクロールバーを使って表示する、より大きな子ボックスを配置したディスプレイボックスを作成する。

戻り値

スクロールボックスオブジェクトへの参照

引数

size(h,v)（オプション）*h*引数および *v*引数で、ボックスのサイズをピクセル単位で指定する。

flexible(Boolean)（オプション）真（1）の場合は、ウィンドウに合わせてボックスのサイズも変更される。偽（0）の場合は、ウィンドウのサイズが変更されても、ボックスのサイズは変わりません。

display box スクロールボックスには任意の数のディスプレイボックスの引数を設定できる。

ノート

スクロールボックスオブジェクトに次のメッセージを送ることによって、背景色を設定できます。

```
<<Set Background Color( {R, G, B} | <color> )
```

横方向（h）と縦方向（v）のスクロールについてブール値のフラグを設定し、スクロールバーを有効（1）または無効（0）にすることができます。所定の方向のスクロールが無効な場合、スクロールボックスはその方向については標準のコンテナ同様の働きをします。

```
<<Set Scrollers (h, v)
```

スクロールのフラグを戻すには、次のメッセージを使用します。

```
<<Get Scrollers
```

スクロールバーのスクローラーの横方向（h）と縦方向（v）の位置（ピクセル単位）を設定するには、次のメッセージを使用します。

```
<<Set Scroll Position (h,v)
```

現在のスクロール位置を戻すには、次のメッセージを使用します。

```
<<Get Scroll Position
```

横方向と縦方向のスクロール範囲の上限位置を戻すには、次のメッセージを使用します。

```
<<Get Scroll Extents
```

例

次の例は、所定の設定のスクロールボックスを含むウィンドウを表示します。

```
win = New Window( "例",
   sb = Scroll Box(
      Size( 150, 75 ),
      List Box(
         {"First Item", "Second Item",
         "Third Item", "Fourth Item",
         "Fifth Item"},
         width( 200 ),
         max selected( 2 ),
         nlines( 6 )
      )
   )
);
win << Set Window Size( 300, 200 );
sb << Set Scrollers( 1, 1 ); // 両方向のスクロールバーを有効化
sb << Set Scroll Position( 0, 20 ); // スクロールバーのスクローラーの位置を設定
```

Sheet Part("title", display box)

説明

引用符付き文字列 *title* をタイトルにして、指定のディスプレイボックス（*display box*）を含んだディスプレイボックスを戻す。

Slider Box(minValue, maxValue, variable, script, <set width(n)>, <rescale slider(min, max)>)

説明

インタラクティブなスライダコントロールを作成する。

戻り値

ディスプレイボックス（スライダボックス）

引数

minValue, maxValue スライダの最小値および最大値となる数値
variable スライダボックスによって設定・変更される変数
script スライダボックスの移動に伴って実行される任意の有効な JSL コマンド
set width(n) スライダボックスの幅をピクセルで指定
rescale slider(l, u) スライダボックスの最大値（max）および最小値（min）をリセットする

ノート

次の例のように、Set Width と Rescale Slider をコマンドとしてスライダオブジェクトに送ることができます。たとえば、次のように実行します。

```
ex = .6;
New Window( "例", mybox = Slider Box( 0, 1, ex, Show( ex ) ) );
mybox << Set Width( 200 ) << Rescale Slider( 0, 5 );
```

Spacer Box(<size(*h*,*v*)>, <color(color)>)

説明

他のディスプレイボックスとの間のスペースを維持するため、または、LineUp Box 内のセルを埋めるために使用されるディスプレイボックスを作成する。

戻り値

ディスプレイボックスへの参照

引数

size(h,v)（オプション）*h* 引数および *v* 引数で、ボックスのサイズをピクセル単位で指定する。
color(color)（オプション）ボックスの色を、引数で指定された JSL カラーに設定する。

String Col Box("title", {"string", ...})

説明

リストされた文字列（*string*）を含んだ列を、表の中に作成する。

String Col Edit Box("title", {"string", ...})

説明

リストされた文字列（*string*）を含んだ列を、表の中に作成する。この関数によって作成された列の文字列は、編集できます。

ノート

データを取得するには、次のメッセージを使用します。

```
data = obj << Get;
```

Tab Box(Tab Page Box("page title 1", <options>, contents of page 1), Tab Page Box("page title 2", <options>, contents of page 2), ...);

説明

（Tab List Box から名称変更。）タブ付きのウィンドウペインを作成する。引数には、タブページの名前とタブページの内容を交互に同数指定します。

ノート

Tab Page Box に送ることのできるメッセージの名称が、以下のように変更になりました。

- Set Title→Title
- Set Tip→Tip
- Set Icon→Icon
- Set Closeable→Closeable

例

```
New Window( "例",
   Tab Box(
      t1 = Tab Page Box( "alpha", Panel Box( "panel", Text Box( "text" ) ) ),
      t2 = Tab Page Box( "beta", Popup Box( {"x", ex = 1, "y", ex = 2} ) ),
   )
);
```

Table Box(display box, ...)

説明

指定のディスプレイボックス（*display box*）を列としたレポートテーブル（表）を作成する。

Text Box("text", <arguments>)

説明

引用符付き文字列で指定されたテキスト（*text*）を含んだボックスを作成する。

引数

<<Justify Text("position") 引用符付き文字列で指定された位置（左、中央、または右）にテキストを整列する。

<<Set Wrap(pixels) テキストを折り返す位置を設定する。

Text Edit Box("text", <arguments>)

説明

引用符付き文字列（*text*）を含む編集ボックスを作成する。

引数

<<Password Style(boolean) パスワード入力用に、テキストを入力すると、アスタリスクを表示する。
<<Set Script テキストが編集された後、指定のスクリプトを実行する。
<<Set Width(pixesl) テキストを折り返す位置を設定する。

Tree Box(<{rootnodes}>, <size(*width*, *height*)>, <MultiSelect>)

説明

Tree Node で構成される階層構造のツリーを表示するボックスを作成する。

引数

{rootnodes} ボックス内の Tree Node() によって作成されるルートノードの名前を指定する。
size(width, height) ボックスの幅と高さを指定する（単位はピクセル）。
MultiSelect ツリー内で複数のアイテムの選択を可能とする。

Tree Node(<data>)

説明

ツリーボックスディスプレイ内に表示するノードを作成する。Tree Node は、親と子の両方のノードに使用されます。

Triangulation(<dt>, X(col1, col1), <Y(Col)>)

説明

与えられたデータ点に対して Delaunay の三角分割を行い、その結果のオブジェクトを戻す。データ点の座標に重複があった場合、それらのデータ点は一つにまとめられ、オプション指定の Y 値には平均が使われる。

例

```
tri = Triangulation(
   X( [0 0 1 1], [0 1 0 1] ),
   Y( [0 1 2 3] )
);
dt = Open( "$SAMPLE_DATA/Cities.jmp" );
tri = Triangulation( X( :X, :Y ), Y( :POP ) );
```

V Center Box(display box)

説明

引数 display box に含まれているすべての子ディスプレイボックスを垂直方向に中央揃えで配置して戻す。

V List Box(<Align("center"|"right")> display box, ...)

説明

ディスプレイボックスを作成し、その中に別のディスプレイボックスを縦に並べて表示する。

引数

Align ディスプレイボックスの位置を、右揃え（right）または中央揃え（center）に指定する。デフォルトでは、中央揃えで配置されます。

display box リストボックスに含める任意の数のディスプレイボックスを指定する。

V Sheet Box(<<Hold(report), display boxes)

説明

引数に指定された複数のディスプレイボックスを、縦方向に配置したディスプレイボックスを戻す。<<Hold() メッセージは、抜粋元となるレポートをどのシートが保持するかを示す。

V Splitter Box(<size(*h*,*v*)>, display box, <arguments>)

説明

引数に指定された複数のディスプレイボックスを縦方向（パネル）に配置し、分割線で仕切ったディスプレイボックスを戻す。ユーザは分割線をドラッグしてパネルのサイズを変更できます。

引数

display box 分割線ボックスには任意の数のディスプレイボックスの引数を設定できる。

ノート

オプションの引数の詳細については、「H Splitter Box(<size(h,v)>, display box, <arguments>)」（78 ページ）を参照してください。

Web Browser Box("url")

説明

Web ページを表示するディスプレイボックスを作成する。Windows でのみ使用できます。

戻り値

Web ブラウザボックスオブジェクトへの参照

引数

url 表示する Web ページの URL を示す引用符付き文字列

例

次の例では、分割線で仕切られたディスプレイボックスを作成し、左に Web ブラウザボックスを、右にバブルプロットを配置します。

```
dt = Open( "$SAMPLE_DATA/PopAgeGroup.jmp" );
New Window( "例",
   H Splitter Box(
      Size( 800, 300 ),
      wb = Web Browser Box( "http://www.jmp.com" ),
      dt << Run Script( "バブルプロット 地域" )
```

```
    )
);
wb << Set Auto Stretching( 1, 1 ); // 自動的に縦横のサイズを調整する
wb << Set Max Size( 10000, 10000 ); // 最大サイズをピクセルで指定
```

Window(<string|int>)

戻り値

開いているウィンドウすべてへの参照のリスト、または指定されたウィンドウへの参照

引数

string 開いている特定のウィンドウの名前を示す引用符付き文字列

int 開いている特定のウィンドウを示す番号

ノート

引数が指定されない場合は、開いているすべてのウィンドウのリストを戻します。

引数で指定されたウィンドウ名または数値が存在しない場合、空のリストを戻します。

Wrap List Box(display box, ...)

説明

リストボックスを作成し、その中に別のディスプレイボックスを横に並べて表示する（印刷時には、折り返して表示する）。

引数

display box リストボックスに含める任意の数のディスプレイボックスを指定する。

式の関数

Arg(expr, i)

Arg Expr(expr, i)

説明

指定された式から *i* 番目の引数を戻す。

戻り値

式 *expr* 内の *i* 番目の引数。

i 番目の引数がない、または指定されていない場合は、Empty() を戻します。

引数

expr JSL スクリプトですでに定義されている式

i どの引数を戻すかを指定する整数

ノート

ArgExpr() は前リリースの JMP での古い形式です。代わりに、Arg() を使用してください。

Eval Expr(expr)

説明

外側の式は未評価のまま残して、*expr* に含まれている内側の式だけをすべて評価する。

戻り値

expr 内のすべての式を評価した結果の式

引数

expr 任意の JSL 式

Expr(x)

説明

引数を評価しないまま戻す（式のクォート）。

戻り値

未評価の引数

引数

x 任意の有効な JSL 式

Extract Expr(expr, pattern)

説明

パターン（*pattern*）に一致する式（*expr*）を見つける。

戻り値

指定した *pattern* とマッチするパターン

引数

expr 任意の式
pattern 任意のパターン

Head(exprArg)

Head Expr(exprArg)

説明

評価された式の一番先頭の関数名を引数を付けないで戻す。

ノート

Head Expr() は廃止されました。代わりに、Head() を使用してください。

Head Name(expr)

Head Name Expr(expr)

説明

式の一番先頭の関数名を文字列として戻します。

ノート

Head Name Expr() は廃止されました。代わりに、Head Name() を使用してください。

Integrate(expr, varname, lowLimit, upLimit, <<Tolerance(1e-10), <<StoreInfo(list), <<StartingValue(val))

説明

Gander and Gautschi (2000) の数値積分によって、1 次元の積分を行う。

引数

expr 被積分関数の式。

varname 積分変数の名前。この変数に値が含まれる場合は、積分の精度を高めるための基準値（初期値）として使用されます。

lowLimit 積分の下限。負の無限大を指定するには、空白にする。

upLimit 積分の上限。正の無限大を指定するには、空白にする。

StoreInfo StoreInfo() の引数に数値計算をした診断が保存される。

StartingValue 積分の精度を高めるための基準値（初期値）。

N Arg(exprArg)

説明

評価した式の先頭部にある引数の個数を戻す。

N Arg Expr(exprArg)

説明

式の先頭部にある引数の個数を戻す。

Name Expr(x)

説明

x に格納されている式を評価せずに戻す。

ファイル関数

Close(<dt|query>, <nosave|save("path")>)

説明

データテーブル、クエリー、または JSON ファイルを閉じる。引数が指定されていない場合は、現在のファイルを閉じます。ファイルが変更されている場合は、自動的に保存します。従属するウィンドウ（データテーブルを元に作成されたレポートウィンドウなど）もすべて閉じられます。

戻り値

なし

引数

dt（オプション）データテーブル、クエリー、または JSON ファイルへの参照。

nosave|save("path")（オプション）ファイルを閉じる前に指定のパスに保存するか、または保存しないで閉じるかを指定する。

Close All(type, <invisible|private>, <noSave|save>)

説明

指定されたタイプ（*type*）のウィンドウをすべて閉じる。

引数

type 閉じるウィンドウのタイプ名を示す名前付き引数。使用できるタイプは、Data Tables（データテーブル）、Reports（レポート）、および Journals（ジャーナル）

invisible（オプション）非表示（invisible）のデータテーブルをすべて閉じる。

private（オプション）プライベート（private）に指定したデータテーブルをすべて閉じる。

noSave または Save（オプション）指定されたタイプのウィンドウを閉じる前に保存するか、保存せずに閉じるかを指定する引数。

Close Log(Boolean)

説明

ログウィンドウを閉じる。

Convert File Path(path, <"absolute"|"relative">, <"posix"|"windows">, <base(path)>)

説明

引数に従ってファイルパスを変換する。

戻り値

変換したパス

引数

path Windows または POSIX のパス名

absolute|relative（オプション）戻り値のパス名を絶対パスにするか相対パスにするかを指定する引用符付き文字列。デフォルト値は絶対パス。

posix|windows（オプション）戻り値のパス名を Windows 形式にするか POSIX 形式にするかを指定する引用符付き文字列。デフォルト値は POSIX。

base(path)（オプション）基準となるパス名を指定する。相対パスを指定している場合に便利。デフォルト値はデフォルトのディレクトリ。

Copy Directory("from path", "to path", <recursive=Boolean>)

説明

ファイルをあるディレクトリから別のディレクトリにコピーする。オプションで、サブディレクトリもコピーできます。引数 *to path* で指定された場所にディレクトリが作成されるので、コピーするディレクトリの名前を引数 *to path* に含めてはいけません。

戻り値

ディレクトリがコピーされた場合は 1、そうでない場合は 0。

引数

from path 新しいディレクトリにコピーするファイルが含まれているディレクトリを指定する。

to path ファイルのコピー先となる新しいディレクトリの作成場所のパスを指定する。

<recursive(Boolean)> コピー元ディレクトリ（*from path*）のサブディレクトリ構造をコピー先（*to path*）にコピーするかどうかを指定する。Windows XP より後にリリースされた Windows オペレーティングシステムでは、常にサブディレクトリ構造がコピーされます。

Copy File("from path", "to path")

説明

ファイルを、元のファイルと同名または別名の新しいファイルにコピーする。

戻り値

ファイルがコピーされた場合は 1、そうでない場合は 0。

引数

from path 新しいファイルにコピーするファイルのパスとファイル名を指定する。

to path 新しいファイルのパスとファイル名を指定する。

Create Database Connection((string, <DriverPrompt(1)>) | "Connect Dialog");

説明

指定したデータベースへの接続を確立するか、ユーザにログイン情報を入力するよう促す。

戻り値

データベース接続のハンドラ。

引数

string データソース名やユーザ名などの情報を含むサーバー接続文字列。

Driver Prompt（オプション）必要に応じて ODBC ドライバが接続情報の入力を促せるようにするためのブール値の引数。

"Connect Dialog"［データソースの選択］ウィンドウを開くための文字列引数。ユーザはこのウィンドウでデータベースを選択する。

例

データソース名、ユーザ名、およびパスワードを指定する例 :

```
Create Database Connection( "dsn=Books;UID=johnsmith;password=Christmas" );
```

接続の文字列にはパスワードを含めず、ユーザに接続情報の入力を促すウィンドウを表示するようODBC ドライバに要求する例 :

```
Create Database Connection( "dsn=Books;UID=johnsmith", Driver Prompt( 1 ) );
```

"Connect Dialog" を指定してユーザによるデータソースの選択を可能とする例 :

```
Create Database Connection( "Connect Dialog" );
```

Create Directory("path")

説明

path 引数に指定した場所に新しいディレクトリを作成する。

戻り値

ディレクトリが作成された場合は 1、そうでない場合は 0。

引数

path 新しいディレクトリの作成場所のパスを指定する。

Creation Date("path")

説明

指定したファイルまたはディレクトリの作成日を戻す。

戻り値

作成日

引数

path 作成日を調べる対象のディレクトリのパスまたはファイル名。

Delete Directory("path")

説明

指定したディレクトリとその内容、およびサブディレクトリを削除する。

戻り値

ディレクトリが削除された場合は 1、そうでない場合は 0。

引数

path 削除するディレクトリのパスとディレクトリ名。

Delete File("path")

説明

指定したファイルを削除する。

戻り値

ファイルが削除された場合は 1、そうでない場合は 0。

引数

path 削除するファイルのパスとファイル名を指定する。

Directory Exists("path")

説明

指定したディレクトリが存在するかどうかを調べる。

戻り値

ディレクトリが存在するときは1、そうでない場合は0

引数

path 調べるディレクトリのパスとディレクトリ名。

File Exists("path")

説明

指定したファイル名が指定したパスに存在するかどうかを調べる。

戻り値

ファイルが存在するときは1、そうでない場合は0

引数

path 調べるファイルのパスとファイル名を指定する。

Files In Directory(path, <"recursive">)

説明

path で指定されたディレクトリ内のファイル名を、リストにして戻す。

戻り値

ファイル名のリスト。"recursive" を指定しなかった場合は、ディレクトリ名がリストに含まれます。

引数

path 有効なパス名

recursive（オプション）このオプションを指定すると、パスにある全フォルダ、およびそれらのフォルダに含まれる全サブフォルダから、ファイルが検索される。

Get Current Directory()

説明

「ファイルを開く」ウィンドウで使用されるカレントディレクトリを取得する。**メモ：**オペレーティングシステムによって、カレントディレクトリがJMPのインストールフォルダ以外の場所に変更されている場合があります。この関数の動作は変更されました。MacintoshおよびWindowsのカレントディレクトリの初期値は、ユーザのDocumentsフォルダです。従来は、Windowsでは jmp.exe が含まれるフォルダ、またはJMPを起動したときにオペレーティングシステムが設定するワーキングフォルダが戻されていました。

– JMPが、JMPショートカットを使って開かれた場合、この関数はJMPのインストールディレクトリを戻す。たとえば、Windowsの場合、カレントディレクトリは "¥C:¥Program Files¥SAS¥JMPPRO¥/13¥" または "¥C:¥Program Files¥SAS¥JMP¥/13¥" です。

– JMPが、JMPファイル（.jmp、.jsl、.jrnなど）をダブルクリックして開かれた場合、このコマンドは開いたファイルのパスを戻します。

– Set Current Directory()関数を使ってカレントディレクトリが設定されている場合は、その指定されたパスを戻します。

「Set Current Directory("path")」（106ページ）を参照してください。

戻り値

文字列で示した絶対パス名

引数

なし

Get Default Directory()

説明

ユーザのデフォルトのディレクトリを取得する。このパスが相対パスに使用されます。

Set Default Directory()関数を使ってデフォルトのディレクトリが設定されている場合、Get Default Directory()がそのSet Default Directory()関数と同じスクリプト内にある限り、その指定されたパスを戻します。

「Set Default Directory("path")」（107ページ）を参照してください。

戻り値

文字列で示した絶対パス名

引数

なし

Get Excel Worksheets("absolute path")

説明

指定されたMicrosoft Excelワークブックに含まれるワークシートのリストを取得する。ワークシートがない場合は、空のリストを戻します。

ノート

この関数は、.xlsxおよびExcel 1997以降のワークブックに対応しています。

Get File Search Path()

説明

現在、ファイルを開く時に検索されるディレクトリの一覧をリストで戻す。

このリストは、Set File Search Path()関数を使って設定されます。「Set File Search Path({path or list of paths})」（107ページ）を参照してください。

戻り値

文字列で示したパス名のリスト

Get Path Variable("name")

説明
name に指定されたパス変数の値を取得する。

戻り値
文字列で示した絶対パス名

引数
name パス変数を示す引用符付き文字列 （例 : SAMPLE_DATA, SAMPLE_SCRIPTS）

Is Directory(path)

説明
path 引数がディレクトリのときは 1、そうでなければ 0 を戻す。

Is Directory Writable(path)

説明
path 引数に指定されたディレクトリが書き込み可能な場合は 1、そうでなければ 0 を戻す。

Is File(path)

説明
path 引数がファイルのときは 1、そうでなければ 0 を戻す。

Is File Writable(path)

説明
path 引数に指定されたファイルが書き込み可能な場合は 1、そうでなければ 0 を戻す。

Is Log Open()

説明
ログウィンドウが開いているかどうかを示すブール値を戻す。

Last Modification Date("path")

説明
指定されたファイルまたはディレクトリの最終更新日を戻す。

戻り値
最終更新日。

引数
path ディレクトリ名またはファイル名を指定する。

Load Text File(path, <arguments>)

説明

指定されたパス（*path*）にあるテキストファイルを、JSL 変数に読み込む。

戻り値

文字列

引数

path テキストファイルを指すパス名。URL でも可

Charset（オプション）文字セットを指定する。以下の引数を使用できます。

- "best guess" 文字セットを推定する。
- force("throw"|"alert"|"silent") （オプション）文字セットを推定できなかった場合の処理を指定する引数。

Line Separator("character")（オプション）行末文字を指定する。たとえば、"\!n"（ラインフィード文字）や、"\!t"（タブ）を指定します。

XMLParse（オプション）XML ファイルを JSL に変換する。

SASODSXML（オプション）SAS の ODS におけるデフォルトの XML 形式として、テキストファイルを解釈する。

BLOB(<arguments>)（オプション）ファイル内のデータを、文字列ではなく BLOB として戻す。ファイルの一部を読み込むには、以下の引数を使用できます。

- ReadOffsetFromBegin(*n*) ファイルの読み込み開始位置を、ファイル先頭からのオフセット値（ゼロベース）で指定する。
- ReadOffsetFromEnd(*n*) ファイルの読み込み開始位置を、ファイル末尾からのオフセット値（ゼロベース）で指定する。
- ReadLength(*n*) ファイルから読み込むバイト数を指定する（ファイルの先頭またはオフセット値から開始）。
- Base64Compressed(0|1) BLOB を印字可能な形式に変換する方法を指定する。0 を指定すると、JMP の ASCII~HEX 表記を使用します（デフォルトかつ推奨）。1 を指定すると、BLOB を base64 形式で圧縮・変換して印字します。

Move Directory("from path", "to path")

説明

ディレクトリとその内容（サブディレクトリも含む）を指定の場所から、もう 1 つ別の指定場所に移動する。

戻り値

ディレクトリが移動した場合は 1、そうでない場合は 0。

引数

from path 移動元ディレクトリのパスとディレクトリ名。

to path 移動先ディレクトリのパスとディレクトリ名。

Move File("from path", "to path")

説明

ファイルを指定のパスから、もう1つ別の指定場所に移動する。ファイル名は同一のままでも変更することもできます。

戻り値

ファイルが移動した場合は1、そうでない場合は0。

引数

from path 移動元ファイルのパスとファイル名を指定する。
to path 移動先ファイルのパスとファイル名を指定する。

ノート

Windowsでは、ファイルを存在しないフォルダに移動しようとすると、フォルダが作成され、1が戻されます。Macintoshでは、フォルダは作成されず、エラーが戻されます。

Open("path", <arguments>)

説明

*path*引数で指定されたファイルから作成されたデータテーブルやその他のJMPファイル、またはオブジェクトを開く。path引数が未指定の場合は、「開く」ウィンドウが表示されます。JSONまたはHDF5ファイルも開けます。具体的なファイルタイプに適用される引数の詳細については、JMPの「スクリプトの索引」の例を参照してください。

引数

Add to Recent Files(Boolean) ホームウィンドウの「最近使ったファイル」リストにファイルを追加するかどうかを指定する。

Charset("option") テキストファイルの読み込みに使用できる文字セットのオプションは、Best Guess、utf-8、utf-16、us-ascii、windows-1252、x-max-roman、x-mac-japanese、shift-jis、euc-jp、utf-16be、およびgb2312。

Column Names Start(n) | Column Names are on line(n) 読み込み対象のテキストファイル内で、列名が入力されている最初の行の番号を指定する。テキストファイルでセル間に改行が使われている場合、列名が複数行に分かれていることがあります。

Columns(colName = colType(colWidth),...) テキストファイル内の列のうち、データテーブルに読み込む対象の列を名前で指定する。

- colName: 読み込まれたデータの列の名前を指定する。
- colType(Character | Numeric): 指定した列を文字タイプとするか、数値タイプとするかを指定する。
- colWidth(n): 指定した列の幅を整数値で指定する。

Columns(<arguments>) ESRIシェープファイル（.shp）の場合、この引数とその設定は以下のようになります。

- Shape=numeric(n): 読み込み対象のESRIシェープファイル内で、シェープ番号が記載されいる列の番号を指定する。

- Part=numeric(n): 読み込み対象のESRIシェープファイル内で、パーツ番号が記載されている列の番号を指定する。
- X=numeric(n): 読み込み対象のESRIシェープファイル内で、10進数表記の経度（±180°の範囲）が記載されている列の番号を指定する。
- Y=numeric(n): 読み込み対象のESRIシェープファイル内で、10進数表記の緯度（±90°の範囲）が記載されている列の番号を指定する。

Compress Allow List Check(Boolean) 読み込んだテキストファイルから作成したデータテーブルをJMPで圧縮するかどうかを指定する。

Compress Character Columns(Boolean) テキストファイルから読み込んだ文字タイプの列をJMPで圧縮するかどうかを指定する。

Compress Numeric Columns(Boolean) テキストファイルから読み込んだ数値タイプの列を圧縮するかどうかを指定する。

Concatenate Worksheets(Boolean) 読み込み対象の複数のExcelワークシートを1つのデータテーブルに連結するかどうかを指定する。

Create Concatenation Column(Boolean) 読み込み対象のExcelファイルの列を1列にまとめるかどうかを指定する。

Data Starts(n) | Data Starts on Line(n) 読み込み対象のテキストファイル内で、データが始まっている行の番号を指定する。

Debug JSL(Boolean) 指定されたJSLスクリプトを開かずに、デバッガ内に開く。

End Of Field(Tab|Space|Comma|Semicolon|Other|None) 読み込み対象のテキストファイルでフィールドの区切り文字として使用されている文字を指定する。複数の文字を指定するには、各文字の指定をカンマで区切ります。"Other" を指定した場合は、EOF Other() 引数で区切り文字を指定します。

End Of Line(CRLF|CR|LF|Semicolon|Other) 読み込み対象のテキストファイルで行の区切り文字として使用されている文字を指定する。複数の文字を指定するには、各文字の指定をカンマで区切ります。"Other" を指定した場合は、EOL Other() 引数で区切り文字を指定します。

EOF Other("char") 読み込み対象のテキストファイルで、End of Field で指定できる文字とは異なるフィールド区切り文字が使用されている場合は、使用する文字をこの引数で指定する。

EOL Other("char") 読み込み対象のテキストファイルで、End of Line で指定できる文字とは異なる行の区切り文字が使用されている場合は、使用する文字をこの引数で指定する。

Excel Wizard Microsoft Excelワークシートを、Excel読み込みウィザードで開く。この引数を指定しないと、ワークシートはそのままデータテーブルとして開かれます。

File Type（オプション）開くファイルの種類を指定する文字列（たとえば、xls、bmp、jpg、またはgif）。この文字列を指定しない場合は、該当するファイルタイプのデフォルトのプログラムでファイルが開きます。

メモ：zip アーカイブの場合は、path 引数を指定します。拡張子（.zip）は必要ありません。zip アーカイブに送るメッセージについては、「JSL メッセージ」章の「Zip アーカイブ」（361 ページ）を参照してください。基本的な機能は、zip アーカイブ内のファイルのリストを取得すること、zip アーカイブ内のファイルを文字列または BLOB に読み込むこと、ファイルを zip アーカイブに書き込むことです。zip アーカイブを読み取ると、内容が一時的にメモリに格納されます。非常に大きな zip アーカイブを読み取ると、エラーが発生する場合があります。

`Force Refresh` 指定された JMP ファイル（.jrn、.jsl、.jrp、または .jmpappsource）を保存せずに閉じ、ディスクから再び開きます。ファイルを最後に開いてから変更していた場合、その変更内容は削除されます。

`HTML Table(n, ColumnNames(n), DataStarts(n))` HTML Web ページからテーブルを読み込むには、URL をファイルパスとして使用する。オプションの引数 *n* で、Web ページ上の開きたいテーブルの番号 *n* を指定します。値を省略した場合、ページ上の最初の表だけが読み込まれます。オプションの引数 `ColumnNames(`*n*`)` で、列名となる行を指定できます。オプションの引数 `DataStarts(`*n*`)` では、データが始まる行を指定できます。

ヒント：読み込もうとしているテーブルにイメージが含まれている場合、それらは最初テキストとして読み込まれます。JMP データテーブルにイメージをロードするには、自動的に生成される Load Pictures というテーブルスクリプトを実行します。すると、イメージを含む新しい式の列が作成されます。式の列の詳細については、『JMP の使用法』の「列情報ウィンドウ」章を参照してください。

`Ignore Columns( "col", ... )` JMP データテーブルまたはその他の JMP ファイル内の列のうち、データテーブルには含めない列の名前を指定する。

`Invisible` ファイルを非表示で開く。この引用符付き引数は、データテーブル、JMP ファイル、外部ファイル、テキストファイル、Excel ファイル、SAS ファイル、ESRI シェープファイル、または HTML ファイルにのみ適用されます。データテーブルは、「JMP ホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。

`Labels(Boolean)` 読み込み対象のテキストファイルの最初の行にラベルや列見出しが含まれているかどうかを指定する。デフォルト値は `1`（真）。

`Lines to Read(n)` テキストファイル内の読み込み対象とする行の数を指定する。JMP では、列名が読み取られた行の次の行から行数のカウントを開始します。

`Number of Columns(n)` 読み込み対象のテキストファイルに含まれている列の数を指定する。

`Run JSL(Boolean)` 指定された JSL ファイルを、開かずに実行する。ブール値またはブール値を含む式を引数として指定します。スクリプトが、スクリプトを自動的に実行する `//!` で始まっている場合、このスクリプトを開くにはブール値（0）を指定します。

`Password( "password" )` パスワードで保護された Microsoft Excel の .xls ファイルや SAS ファイルのパスワードを指定する。ここで指定しておくと、手動で入力する必要がなくなります。パスワードは暗号化されません。（パスワードで保護された Microsoft Excel の .xlsx ファイルは読み込むことができません。）

`Private` データテーブルを非表示で開き、JMP ホームウィンドウにも［ウィンドウ］メニューにもテーブル名を表示しない。ユーザが見る必要のない一時的なデータテーブルを作成したいときに使用してください。この引用符付き引数はデータテーブル、JMP ファイル、外部ファイル、テキストファイル、

Excel ファイル、SAS ファイル、ESRI シェープファイル、または HTML ファイルにのみ適用されます。

Scan Whole File(Boolean) 列のデータタイプを判断するためにどれくらい JMP がテキストファイルをスキャンするかを指定する。値はブール値。デフォルト値の 1（真）では、データタイプが判断されるまでファイル全体がスキャンされます。大きなファイルを読み込む場合は、この値を 0（偽）に設定することを検討してください。その場合、ファイルは 5 秒間スキャンされます。

Select Columns("col", ...) JMP データテーブルまたはその他の JMP ファイル内の列のうち、データテーブルに含める列の名前を指定する。

Strip Quotes | Strip Enclosing Quotes (Boolean) テキストファイル内のフィールドが引用符で囲まれているとき、この引数が真の場合は引用符を削除し、偽の場合は引用符を削除しない。デフォルト値は 1（真）。

Table Contains Column Headers (Boolean) 読み込み対象のテキストファイルの最初の行にラベルや列見出しが含まれているかどうかを指定する。デフォルト値は 1（真）。

Text Wizard テキストファイルを、テキスト読み込みプレビューで開く。このウィンドウで、読み込みオプションを選択できます。このオプションを指定しない場合は、JMP の環境設定の［テキストデータファイル］の設定に従って、データテーブルとして自動的に読み込まれます。

Treat Empty Columns as Numeric(Boolean) テキストファイル内の欠測値データの列を文字タイプではなく数値タイプとするかどうかを指定する。ピリオド、Unicode のドット、NaN、空白文字列は欠測値と判断されます。デフォルト値は 0（偽）。

Use Apostrophe as Quotation Mark(Boolean) テキストファイルの読み込み時にアポストロフィを引用符として使用することを宣言する。テキストデータが標準的な形式ではなく、フィールドが引用符ではなくアポストロフィで囲まれている場合以外、このオプションは推奨されません。デフォルト値は 0（偽）。

Use Labels for Var Names(Boolean) SAS データセットの場合、SAS ラベルを JMP の列名として使用するかどうかを指定する。デフォルト値は 0（偽）。

Use for all sheets(Boolean) Worksheets の設定を、データテーブルとして開く Excel ファイルのすべてのワークシートで使用するかどうかを指定する。

Worksheet Settings(Boolean, <options>) Excel ファイルの JMP データテーブルへの読み込みオプションを指定する。次のいずれかのオプションを指定できます。

- Has Column Headers(Boolean): Excel ファイルの第 1 行目が列見出しであることを示す。
- Number of Rows in Headers(n): Excel ファイル内で列見出しとして使用されている行の数を指定する。
- Headers Start on Row(n): Excel ファイル内で列見出しが入力されている最初の行の番号を指定する。デフォルト値は 1 です。
- Data Starts on Row(n): Excel ファイル内でデータが入力されている最初の行の番号を指定する。
- Data Starts on Column(n): Excel ファイル内でデータが入力されている最初の列の番号を指定する。
- Data Ends on Row(n): Excel ファイル内でデータが入力されている最後の行の番号を指定する。
- Data Ends on Column(n): Excel ファイル内でデータが入力されている最後の列の番号を指定する。

- Replicated Spanned Rows(Boolean): Excelファイル内にあるセルが結合された行のデータを複製して読み込むかどうかを指定する。
- Suppress Hidden Rows(Boolean): Excelファイルで非表示になっている行をJMPに読み込まないかどうかを指定する。
- Suppress Hidden Columns(Boolean): Excelファイルで非表示になっている列をJMPに読み込まないかどうかを指定する。
- Treat as Hierarchy(Boolean): JMPにExcelファイルを読み込むときに、複数の列見出しを階層として扱うかどうかを指定する。真の場合、Excelファイルは、複数の見出し行が名が階層構造になった状態で読み込まれます。

Worksheets ("sheet name"|{"sheet name", "sheet name",...}|"*n*") 名前が指定されたワークシート、名前リスト内のすべてのワークシート、またはインデックス番号が指定されたワークシートを開く。ワークシートが未指定の場合は、スプレッドシート内のすべてのワークシートが個別のデータテーブルとして開きます。

Year Rule | Two digit year rule
("1900-1999"|"1910-2009"|"1920-2019"|"1930-2029"|"1940-2039"|
"1950-2049"|"1960-2059"|"1970-2069"|"1980-2079"|"1990-2089"|"2000-2099") 読み込み対象のテキストファイルで使用されている2桁の年の範囲を指定する。たとえば、一番古い日付が1979年の場合は、"1970-2069" と指定します。

Open Database("connectInfo" "sqlStatement", "outputTableName")

説明

接続情報（*connectInfo*）で指定されたデータベースを、SQL ステートメント（*sqlStatement*）を使って開き、指定の名前（*outputTableName*）をつけたデータテーブルを戻す。

Pick Directory(<"prompt">, <path>, <Show Files>)

説明

ユーザにディレクトリの選択を促し、ディレクトリパスを文字列で戻す。

戻り値

ユーザが選択したディレクトリのパス

引数

prompt（オプション）引用符付き文字列。指定した文字列は、Windows 上の「参照」ウィンドウの最上部に表示されます。

path（オプション）ディレクトリの選択ウィンドウで最初に開く場所を、引用符付き文字列で指定する。

Show Files（ブール値）ディレクトリの選択ウィンドウにファイルを表示するには 1 を、非表示にするには 0 を指定する。デフォルトは 0。

Pick File(<"prompt">, <"initial directory">, <{filter list}>, <first filter>, <save flag>, <"default file">), <multiple>

説明

「開く」ウィンドウを表示し、ユーザにファイルの選択を促す。

戻り値

ユーザが選択したファイルのパス

引数

`prompt`（オプション）引用符付き文字列。指定した文字列は、Windows 上の「開く」ウィンドウの最上部に表示されます。

`initial directory`（オプション）フォルダへの有効なファイルパスである引用符付き文字列。指定したパスは、ファイルを開くウィンドウの開始ディレクトリとして使用されます。これを指定しない場合、または空の文字列を指定した場合は、JMP のデフォルトのディレクトリが使用されます。

`filter list`（オプション）ファイルを開くウィンドウに表示するファイルの種類のリストである引用符付き文字列。構文については、次の例を参照してください。

`first filter`（オプション）フィルタリスト内のどのフィルタを最初に使用するかを示す整数。大きすぎる整数（項目数が 3 のリストに対して 4 など）または小さすぎる整数を指定した場合、リスト内の最初のフィルタが使用されます。

`save flag`（オプション）「開く」ウィンドウと「名前を付けて保存」ウィンドウのどちらを使用するかを指定するブール値。0 を指定した場合、ユーザは JMP で開くファイルを選択できます。1 を指定した場合、ユーザは、選択した種類の新しい空のファイルを、選択したフォルダに保存できます。デフォルト値は 0 です。

`default file` デフォルトでウィンドウに表示されるファイルの名前

`multiple`（オプション）`save flag` が 0 の場合に、ユーザは複数のファイルを選択できます。

ノート

すべての引数はオプションですが、位置に依存します。たとえば、caption（キャプション）と、initial directory（最初のディレクトリ）の両方を指定しないと、filter list（フィルタリスト）を指定することはできません。

コンピュータの物理メモリ内のバッファサイズは、ユーザが開くことのできるファイルの数に影響します。

例

次のスクリプトは、ウィンドウのタイトルを「**JMP ファイルの選択**」としたファイルを開くウィンドウにおいて、JMP の「Samples/Data」ディレクトリを開き、ファイルの種類のリストに「**JMP ファイル**」と「**すべてのファイル**」を含め、そのうち「**JMP ファイル**」が選択された状態とし、ファイル名のフィールドに「Hollywood Movies.jmp」を表示します。

```
Pick File(
   "JMP ファイルの選択 ",
   "$SAMPLE_DATA",
   {"JMP Files|jmp;jsl;jrn", "All Files|*"},
   1,
   0,
   "Hollywood Movies.jmp"
);
```

Rename Directory("old path name", "new directory name")

説明

ディレクトリを移動またはコピーせずに名前を変更する。

戻り値

ディレクトリの名前を変更した場合は 1、そうでない場合は 0。

引数

`old path name` 変更前のディレクトリのパスと名前を指定する。
`new name` 新しいディレクトリ名を指定する。

ノート

*新しいディレクトリ名*を指定するときは、フルパスではなくディレクトリ名だけを含めます。

Rename File("old path name", "new name")

説明

ファイルを移動またはコピーせずに名前を変更する。

戻り値

ファイル名を変更した場合は 1、そうでない場合は 0。

引数

`old path name` 変更前のファイルのパスと名前を指定する。
`new name` 新しいファイル名を指定する。

ノート

*新しい名前*を指定するときは、フルパスではなくファイル名だけを含めます。

Save Text File(path, text)

説明

JSL 変数 text を path で指定したファイルに保存する。

Set Current Directory("path")

説明

「ファイルを開く」ウィンドウで参照されるカレントディレクトリを設定する。**メモ：**オペレーティングシステムによって、カレントディレクトリが JMP のインストールフォルダ以外の場所に変更されている場合があります。この関数は将来廃止または変更される可能性があります。互換性のために新しい値を設定したり戻すことは行いますが、設定された値は使用されません。

ファイル処理のたびにカレントディレクトリが設定されてしまうのを防ぐには、ファイルパスの文字列を定義し、それをファイル名に連結します。詳細については、『スクリプトガイド』の「パス変数」の節を参照してください。

「Get Current Directory()」（96 ページ）を参照してください。

Set Default Directory("path")

説明

デフォルトのディレクトリを設定する。以降、このディレクトリが相対パスの基準となります。

「Get Default Directory()」（97 ページ）を参照してください。

Set File Search Path({path or list of paths})

説明

ファイルを開く時に検索されるディレクトリを設定する。パスとして {"."} を設定すると、カレントディレクトリが使用されます。

「Get File Search Path()」（97 ページ）を参照してください。

例

```
Set File Search Path( {"C:¥JMP¥13¥source", "C:¥Program
   Files¥SAS¥JMPPRO¥13¥Samples"} );
```

Set Path Variable("name")

説明

パス変数 (name) に、パス (path) を格納する。

TripleS Import("path", <arguments>)

説明

指定された Triple-S（SSS）形式のファイル（調査ファイル）を読み込む。SSS 形式は、.xml または .sss のいずれかと、.csv、.dat、または .asc のいずれかのファイルのペアで構成されます。ペアのファイルは拡張子以外同じ名前で、同じフォルダになければなりません。

引数

`path` xml または sss ファイルのフルパスを指定する引用符付き文字列。

`Invisible`（オプション）テーブルを非表示にする。データテーブルは、「JMP ホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。非表示のデータテーブルは、ユーザによって明示的に閉じられるまでメモリ内に保管され、その分 JMP が使用できるメモリが減ることになります。非表示のテーブルを明示的に閉じるには、`Close(dt)` を実行します。ここで、*`dt`* は `TripleS Import` 関数から戻されたデータテーブル参照の変数です。

`Use Labels for Imported Column Names`（オプション）ブール値。ラベル名を列見出しとして使用します。デフォルト値は `1`（真）。

例

```
dt = TripleS Import( "C:/Data/airlines.sss", Invisible, Use Labels for Imported
   Column Names( 0 ) );
```

財務関数

Double Declining Balance(cost, salvage, life, period, <factor>)

説明

指定の期における資産の減価償却費を戻す。倍額定率法または他の償却率によって、減価償却費を計算します。

引数

cost 初期費用
salvage 減価償却終了後の価値
life 寿命。減価償却の期間
period 減価償却費を求めたい期。寿命と同じ単位で指定してください。
factor（オプション）償却率を示す数値。デフォルト値は 2 です。

ノート

この関数は、Excel の DDB 関数に相当します。

Future Value(rate, nper, pmt, <pv>, <type>)

説明

利率が一定な状況で定期に定額支払をした場合の、投資の将来価値を戻す。

引数

rate 利率
nper 投資の期間
pmt 定額支払における毎回の支払額
pv（オプション）現在価値を示す数値。デフォルト値は 0 です。
type（オプション）期末払いの場合は 0、期首払いの場合は 1 に設定する。デフォルト値は 0 です。

ノート

この関数は、Excel の FV 関数に相当します。

Interest Payment(rate, per, nper, pv, <fv>, <type>)

説明

利率が一定な状況で定期に定額支払をした場合の、支払いにおける利子額を戻す。

引数

rate 利率
per 元金支払額を求める期
nper 投資の期間
pv 現在価値
fv（オプション）将来価値。デフォルト値は 0 です。
type（オプション）期末払いの場合は 0、期首払いの場合は 1 に設定する。デフォルト値は 0 です。

ノート
この関数は、Excel の IPMT 関数に相当します。

Interest Rate(nper, pmt, pv, <fv>, <type>, <guess>)

説明
投資の 1 期あたりの利率を戻す。

引数
nper 投資の期間
pmt 定額支払における毎回の支払額
pv 現在価値
fv（オプション）将来価値。デフォルト値は 0 です。
type（オプション）期末払いの場合は 0、期首払いの場合は 1 に設定する。デフォルト値は 0 です。
guess（オプション）予想される大まかな結果。デフォルトの数は 0.1（10%）です。

ノート
この関数は、Excel の RATE 関数に相当します。

Internal Rate of Return(values, <guess>)

Internal Rate of Return(guess, value1, value2, ...)

説明
一連の定期的なキャッシュフロー（*values*）に対する内部収益率を戻す。

引数
values 値を含んだベクトル（行列）。ただし、各値を、1 つ 1 つの引数に指定する形式でも指定できます。
guess 予測されるおおよその結果を示す数値。デフォルトの数は 0.1（10%）です。

ノート
この関数は、Excel の IRR 関数に相当します。

Modified Internal Rate of Return(values, finance rate, reinvest rate)

Modified Internal Rate of Return(finance rate, reinvest rate, value1, value2, ...)

説明
一連の定期的なキャッシュフローに対する修正内部収益率を戻す。計算において、投資コストと、現金の再投資によって得た利子の両方を考慮します。

引数
values 値を含んだベクトル（行列）。ただし、各値を、1 つ 1 つの引数に指定する形式でも指定できます。
finance rate キャッシュフローの現金に対して支払う利率
reinvest rate キャッシュフローの現金を再投資した場合に受け取る利率

ノート
この関数は、Excel の MIRR 関数に相当します。

Net Present Value(rate, values)

Net Present Value(rate, value1, value2, ...)

説明
投資の正味現在価値を戻す。割引率、および、一連の将来の支払（負の値）や収入（正の値）から計算されます。

引数
rate 割引率
values 値を含んだベクトル（行列）。ただし、各値を、1つ1つの引数に指定する形式でも指定できます。

ノート
この関数は、Excel の NPV 関数に相当します。

Number of Periods(rate, pmt, pv, <fv>, <rate>)

説明
利率が一定な状況で定期に定額支払をした場合の、投資期間数を戻す。

引数
rate 利率
pmt 定額支払における毎回の支払額
pv 現在価値
fv（オプション）将来価値。デフォルト値は 0 です。
type（オプション）期末払いの場合は 0、期首払いの場合は 1 に設定する。デフォルト値は 0 です。

ノート
この関数は、Excel の NPER 関数に相当します。

Payment(rate, nper, pv, <fv>, <type>)

説明
利率が一定な状況で定期に定額支払をした場合の、ローンの支払額を戻す。

引数
rate 利率
nper 投資の期間
pv 現在価値
fv（オプション）将来価値。デフォルト値は 0 です。
type（オプション）期末払いの場合は 0、期首払いの場合は 1 に設定する。デフォルト値は 0 です。

ノート
この関数は、Excel の PMT 関数に相当します。

Present Value(rate, nper, pmt, <fv>, <type>)

説明

投資の現在価値を戻す。

引数

rate 1期あたりの利率

nper 投資の期間

pmt 定額支払における毎回の支払額

fv（オプション）将来価値。デフォルト値は0です。

type（オプション）期末払いの場合は0、期首払いの場合は1に設定する。デフォルト値は0です。

ノート

この関数は、Excelの PV 関数に相当します。

Principal Payment(rate, per, nper, pv, <fv>, <type>)

説明

利率が一定な状況で定期に定額支払をした場合の、支払いにおける元金分を戻す。

引数

rate 1期あたりの利率

per 元金支払額を求める期

nper 投資の期間

pv 現在価値

fv（オプション）将来価値。デフォルト値は0です。

type（オプション）期末払いの場合は0、期首払いの場合は1に設定する。デフォルト値は0です。

ノート

この関数は、Excelの PPMT 関数に相当します。

Straight Line Depreciation(cost, salvage, life)

説明

ある期における減価償却費を戻す。この関数は、定額法によって減価償却費を計算します。

引数

cost 資産の初期費用

salvage 減価償却終了後の価値

life 寿命。減価償却の期間

ノート

この関数は、Excelの SLN 関数に相当します。

Sum Of Years Digits Depreciation(cost, salvage, life, per)

説明

指定の期における減価償却費を戻す。この関数は、級数法によって減価償却費を計算します。

引数

cost 資産の初期費用
salvage 減価償却終了後の価値
life 寿命。減価償却の期間
per 減価償却費を求めたい期。寿命と同じ単位で指定してください。

ノート

この関数は、Excel の SYD 関数に相当します。

グラフ関数

Add Color Theme({"name", <flags>, {color}, <{position}>)

説明

マーカー、データテーブルの行、ツリーマップなどのコンポーネントに適用できる独自のカラーテーマを作成する。Preferences() の中に Add Color Theme(...) を含めることにより、JMP 環境設定にカラーテーマを追加できます。

戻り値

なし

引数

name カラーテーマの名前。
flags（オプション）連続変数またはカテゴリカル変数のカラーテーマリストおよびカラーのカテゴリのためのフラグ。

```
Continuous, <Continuous>, Sequential
Continuous, <Continuous>, Diverging
Continuous, <Continuous>, Chromatic
Categorical, <Continuous>, Sequential
Categorical, <Continuous>, Diverging
Categorical, Qualitative
Categorical, <Continuous>, Chromatic
```

デフォルトのJMPカラーテーマでは、順次（Sequential）は左から右または右から左へ徐々に色が変化します。発散（Diverging）は中央が薄めです。色彩（Chromatic）は明るい色のブロック（グラデーション）で構成されます。定性（Qualitative）以外のすべてのカテゴリは、連続変数とカテゴリカル変数の両方に使用できます。

このフラグを指定しなかった場合、色は連続変数の順次、およびカテゴリカルの順次カテゴリで表示されます。

color RGB 値のリスト。これらの値により、カテゴリカルのカラーテーマ内のブロックと連続変数のカラーテーマ内のグラデーションが定義されます。RGB 値の各リストは、それぞれ環境設定のカラーテーマウィンドウ内のスライダに対応しています。
position（オプション）各色の位置を示す 0 ～ 1 の数字のリスト。各位置は、それぞれ環境設定のカラーテーマウィンドウ内のスライダに対応しています。位置を指定しなかった場合、スライダは均等に配置されます。

例

次の例は、「青 -> 紫」という名前の連続変数のカラーテーマを作成します。色は発散（Diverging）カテゴリです。RGB 値は、4 つのリストで指定されます。

```
Add Color Theme(
   {"青 -> 紫 "", {"Continuous", "Diverging"}, {{0, 0, 255},
   {57, 108, 244}, "white", {128, 0, 100}}} );
```

ノート

定性（Qualitative）以外のすべてのカテゴリは、連続変数とカテゴリカル変数の両方に使用できます。たとえば、「寒色 -> 暖色」の発散テーマは、連続変数とカテゴリカルの両方のテーマリストにあります。JMP で例を見るには、[**環境設定**] > [**グラフ**] を選択してください。

カラーテーマを削除するには、`Remove Color Theme("name")` を使用します。

Arc(x1, y1, x2, y2, startangle, endangle)

説明

引数によって指定された長方形の中に内接する弧を描く。

戻り値

なし

引数

`x1, y1` 長方形の左上隅の座標

`x2, y2` 長方形の右下隅の座標

`startangle, endangle` 度単位の開始角度と終了角度で、*`startangle`* から *`endangle`* まで時計回りに円弧が描画されます。このとき、0 度は時計の 12 時の位置とします。

Arrow(<pixellength>, {x1, y1}, {x2, y2})

説明

指定された点の間に、順に矢印を引いていく。オプションの第 1 引数は、矢じりの長さをピクセルで指定します。

戻り値

なし

引数

`pixellength`（オプション）矢じりの長さをピクセルで指定

`{x1, y1}, {x2, y2}` グラフ内の座標を示す 2 つの数値のリスト

ノート

x 座標が入った行列と、y 座標が入った行列で指定することもできます。`Arrow(<pixellength>, [x1, x2], [y1, y2])`

Back Color("name")

説明

グラフの背景の色を設定する。

戻り値
なし

引数
name 引用符付き色名または色番号（たとえば、赤の場合は "red" または "3"）。

Char To Path("path")

説明
パスの指定を文字形式から行列形式に変換する。

戻り値
行列

引数
path パス名を示す文字列

Circle({x, y}, radius|PixelRadius(n), <...>, <"fill">)

説明
{x,y} を中心として、指定された半径の円を描く。

戻り値
なし

引数
{x, y} グラフ内の座標を示す数値
radius 円の半径。この半径は、縦軸の目盛りに基づくものです。縦軸のサイズを変更すると、円のサイズも変わります。
PixelRadius(n) 円の半径をピクセルで表した数値。このオプションで半径をした場合、縦軸のサイズを変更しても、円のサイズは**変わりません**。
"fill"（オプション）文字列。関数内に定義された円はすべて、現在 fill color で指定されている色で塗りつぶされます。"fill" が省略された場合、円は塗りつぶされません。

ノート
中心点と半径は、任意の順序で指定できます。また、中心点と半径を複数指定すれば、1 つの関数によって複数の円を作成できます。中心点を 1 つ、半径を複数指定した場合、結果は同心円になります。中心点を複数指定した場合、それまでの円がすべて描かれた後に、最後に指定された半径の円が追加されます。例として、次のような指定をしたとします。

```
graphbox(circle({20, 30}, 5, {50, 50}, 15))
```

この場合、2 つではなく、3 つの円が描かれます。まず、(20, 30) を中心とした半径 5 の円、次に (50, 50) を中心とした半径 5 の円、最後に (50, 50) を中心とした半径 15 の円が描かれます。

Color To HLS(color)

説明
色（*color*）の引数（JMP の色番号を含む）を HLS 値のリストに変換する。

戻り値

指定した色（*color*）の色調（H）、明度（L）、彩度（S）の成分を表したリスト。値の範囲は 0 ～ 1 です。

引数

color JMP の色番号

例

ColorToHLS() の結果は、1 つのリスト変数または 3 つのスカラー変数のリストに割り当てることができます。

```
hls = Color To HLS( 8 );
{h, l, s} = Color To HLS( 8 );
Show( hls, h, l, s );
  hls = {0.778005464480874, 0.509803921568627, 0.976};
  h = 0.778005464480874;
  l = 0.509803921568627;
  s = 0.976;
```

Color To RGB(color)

説明

色（*color*）の引数（JMP の色番号を含む）を RGB 値のリストに変換する。

戻り値

指定した色（*color*）の赤（R）、緑（G）、青（B）の成分を表したリスト。値の範囲は 0 ～ 1 です。

引数

color JMP の色番号

例

ColorToRGB() の結果は、1 つのリスト変数または 3 つのスカラー変数のリストに割り当てることができます。

```
rgb = Color To RGB( 8 );
{r, g, b} = Color To RGB( 8 );
Show( rgb, r, g, b );
  rgb = {0.670588235294118, 0.0313725490196078, 0.988235294117647};
  r = 0.670588235294118;
  g = 0.0313725490196078;
  b = 0.988235294117647;
```

Contour(xVector, yVector, zGridMatrix, zContour, <zColors>)

説明

指定のグリッド値で等高線を描く。

戻り値

なし

引数

xVector zGridMatrix を表す *n* 個の値

yVector zGridMatrix を表す *m* 個の値

zGridMatrix 曲面を表す *n* x *m* 行列
zContour（オプション）等高線の値の定義
zColors（オプション）等高線に使用する色の定義

Contour Function(expr, xName, yName, z, < <<XGrid(min, max, incr)>, < <<YGrid(min, max, incr)>, < <<zColor(color)>, < <<zLabeled>, < <<Filled>, < <<FillBetween>, < <<Ternary>, < <<Transparency(alpha|vector))

説明

2 変数に対する関数値の等高線図を描く。等高線の高さを指定する引数（*z*）は 1 つの数値でも行列でもかまいません。

戻り値

なし

引数

expr 任意の式 たとえば、Sine(y)+Cosine(x)
xName、yName 2 変数の名前
z z 値または z 値の行列

オプションの引数

<<XGrid、<<YGrid グリッドの細かさを定義する名前付きオプション
<<zColor 色を定義する数値、もしくは行列。引数はスカラーでも行列でもかまいませんが、数値でなければなりません。
<<zLabeled 等高線にラベルをつける。
<<Filled 現在、zColor で指定されている色で、等高線以上の領域を塗りつぶす。
<<FillBetween 現在、zColor で指定されている色で、隣接する等高線間の領域を塗りつぶす。nz 本の等高線が描かれている場合、このオプションは (nz-1) 個の領域を塗りつぶします。<<Filled オプションでなく、このオプションを使用することをお勧めします。
<<Ternary 三角図内の三角座標系からはみ出る線を切り取る。
<<Transparency 塗りの透明度を設定する。0 ～ 1 の数値のベクトルで指定してください。等高線の各領域を塗りつぶすのに、指定された透明度が循環して使われます。このオプションは、<<FillBetween オプションと組み合わせて使用する場合にのみ、使用してください。

Drag Line(xMatrix, yMatrix, <dragScript>, <mouseupScript>)

説明

引数の行列（xMatrix、yMatrix）で与えられた座標上に、頂点がドラッグ可能な線分を描く。

戻り値

なし

引数

xMatrix *x* 座標の行列
yMatrix *y* 座標の行列
dragScript 任意の有効な JSL スクリプト。ドラッグによって実行される
mouseupScript 任意の有効な JSL スクリプト。マウスボタンを放したときに実行される

Drag Marker(xMatrix, yMatrix, <dragScript>, <mouseupScript>)

説明

引数の行列（xMatrix、yMatrix）で与えられた座標上に、ドラッグ可能なマーカーを描く。

戻り値

なし

引数

xMatrix *x* 座標の行列

yMatrix *y* 座標の行列

dragScript 任意の有効な JSL スクリプト。ドラッグによって実行される

mouseupScript 任意の有効な JSL スクリプト。マウスボタンを放したときに実行される

Drag Polygon(xMatrix, yMatrix, <dragScript>, <mouseupScript>)

説明

引数の行列（xMatrix、yMatrix）で与えられた座標上に、頂点がドラッグ可能な多角形を描く。

戻り値

なし

引数

xMatrix *x* 座標の行列

yMatrix *y* 座標の行列

dragScript 任意の有効な JSL スクリプト。ドラッグによって実行される

mouseupScript 任意の有効な JSL スクリプト。マウスボタンを放したときに実行される

Drag Rect(xMatrix, yMatrix, <dragScript>, <mouseupScript>)

説明

引数の行列（xMatrix、yMatrix）で与えられた座標上に、頂点がドラッグ可能な塗りつぶされた長方形を描く。

戻り値

なし

引数

xMatrix 2 つの *x* 座標の行列

yMatrix 2 つの *y* 座標の行列

dragScript 任意の有効な JSL スクリプト。ドラッグによって実行される

mouseupScript 任意の有効な JSL スクリプト。マウスボタンを放したときに実行される

ノート

xMatrix と *yMatrix* には、値を 2 つずつ指定する必要があります。指定する座標ペアは、rect() を描くための規則に従っていなければなりません。最初の点（*xMatrix* の最初の値と *yMatrix* の最初の値）は、長方形の左上の点となります。2 番目の点（*xMatrix* の 2 番目の値と *yMatrix* の 2 番目の値）は、長方形の右下の点となります。

Drag Text(xMatrix, yMatrix, "text", <dragScript>, <mouseupScript>)

説明

引数の行列（xMatrix、yMatrix）で与えられた座標上に、テキスト（text）を描く。または、リストが指定されている場合は、そのリストの項目を描く。

戻り値

なし

引数

xMatrix *x* 座標の行列

yMatrix *y* 座標の行列

text 引用符付き文字列。グラフに描画されるテキストとなる

dragScript 任意の有効な JSL スクリプト。ドラッグによって実行される

mouseupScript 任意の有効な JSL スクリプト。マウスボタンを放したときに実行される

Fill Color(n)

説明

領域を塗りつぶすときの色を設定する。

戻り値

なし

引数

n 色番号、または、引用符付き文字列で示した色名

Fill Pattern()

説明

塗りつぶし領域のパターンを設定する。例については、『スクリプトガイド』の「スクリプトによるグラフ作成」章を参照してください。

Gradient Function(zexpr, xname, yname, [zlow, zhigh], zcolor([colorlow, colorhigh]), < <<XGrid(min, max, incr)>, < <<YGrid(min, max, incr)> < <<Transparency(alpha|vector))

説明

2 変数に対する関数値を色で描いたグラフを作成する。グリッド上の長方形を、式の値に基づき、小さい値に対する色と大きい値に対する色を混合して塗りつぶす。

例

```
Gradient Function(Log(a * a + b * b),
a, b, [2 10],
Z Color([4, 6]));
```

この例において、*Zexpr* に指定された式は、変数 a および b で表された関数になっています。*zlow* から *zhigh* までの値（2 ～ 10）で、色の混合が行われます。*Zcolor* は混合される 2 つの色を定義します（4 は緑、6 はオレンジ）。

H Line(<x1, x2>, y)

説明

y の位置に横線を描く。*x* 座標の開始点および終了点（*x1* および *x2*）が指定された場合、*y* の位置に *x1* から *x2* まで線を引きます。引数 *y* に行列を指定し、複数の線を引くこともできます。

H Size()

説明

グラフフレームの横のサイズをピクセル単位で戻す。

Handle(a, b, dragScript <,mouseupScript>)

説明

引数（*a* と *b*）で与えられた座標に、ドラッグ可能なマーカーを配置する。最初のスクリプト（dragScript）はドラッグした際に、2 番目のスクリプト（mouseupScript）はマウスのボタンを放したときに実行されます。

Heat Color(n, <"color theme">)

説明

指定されたカラーテーマ（"*theme*"；彩色方法）において、*n* に対応する色の値を戻す。色の値は、JMP で定義されている色の番号で戻されます。

戻り値

JMP の色番号。整数

引数

n 0 ～ 1 までの数値。

theme セルプロットでサポートされている任意の引用符付きカラーテーマ名。デフォルト値は「青 -> グレー -> 赤」（Blue to Gray to Red）。

HLS Color(h, l, s)

HLS Color({h, l, s})

説明

指定された色調（h）、明度（l）、および彩度（s）の値を JMP の色番号に変換する。

戻り値

JMP の色番号。整数

引数

色調、明度、および彩度、またはそれら 3 つの値を含んだリスト。指定できる値の範囲は 0 ～ 1

In Path(x, y, path)

説明

x と *y* で指定された点が *path* 内にあるかどうかを特定する。

戻り値

点 (x, y) が指定されたパス内にあれば真（1）、そうでない場合は偽（0）を戻す。

引数

x、y 点の座標

path パスを示す行列または文字列

In Polygon(x, y, xx, yy)

In Polygon(x, y, xyPolygon)

説明

指定の点（*x,y*）が、ベクトル引数（*xx* と *yy*）で定義された多角形の内側にあるかどうかを示すブール値（1 または 0）を戻す。

ベクトル引数（*xx,yy*）の代わりに、2 列の行列（*xyPolygon*）を使用することもできます。また、*x* と *y* として、互いに対応したベクトルを指定することもでき、その場合、各（*x*、*y*）のペアが指定された多角形の中に入るかどうかに基づき、0 と 1 から構成されるベクトルが戻されます。

Level Color(i, <n>, <"Color Theme">)

説明

グラフ内のカテゴリカルデータに JMP の色番号を割り当てる。

戻り値

JMP の色番号。整数

引数

i 1 以上、かつ、n で指定されたカテゴリ数以下の整数。

n カテゴリ数

"Color Theme" ［列プロパティ］の「値の色」リストに表示されているカラーテーマのいずれか。指定されていない場合、JMP のデフォルトのカラーテーマが適用されます

ノート

第 2 引数が数値ではなく文字列の場合、第 2 引数はカラーテーマとみなされます。

Line({x1, y1}, {x2, y2}, ...), <<ValueSpace(0|1)

Line([x1, x2, ...], [y1, y2, ...]), <<ValueSpace(0|1)

説明

点間に線を描く。

引数

{x1, y1}, {x2, y2} または [x1, x2, ...], [y1, y2, ...] 点の座標を示すペアを含むリスト、または x の行列と y の行列

<<ValueSpace (Boolean) バブルプロットの軌跡のように、線がデータの変化を表している場合、その軌跡の線を描く。Boolean には定数または式を指定できます。

Line Style(n)

説明

グラフの線種を設定する。

引数

n 線種名または線種の番号

- 0／Solid（実線）
- 1／Dotted（点線）
- 2／Dashed（破線）
- 3／DashDot（一点鎖線）
- 4／DashDotDot（二点鎖線）

Marker(<markerState>, {x1, y1}, {x2, y2}, ...)

Marker(<markerState>, [x1, x2, ...], [y1, y2, ...})

説明

リストまたは行列で指定された座標にマーカーを描く。オプションの引数 *markerState* でマーカーの種類を設定できます。

Marker Size(n)

説明

マーカーのサイズを指定する。

Mousetrap(dragscript, mouseupscript)

説明

クリックの座標を読み取り、グラフのプロパティを更新する。最初のスクリプト（dragScript）はドラッグした際に、2 番目のスクリプト（mouseupScript）はマウスのボタンを放したときに実行されます。

Normal Contour(prob, meanMatrix, stdMatrix, corrMatrix, <colorsMatrix>, <fill=x>)

説明

k 個の母集団の、2 変量正規分布の等高線を描く。

引数

prob 累積確率。スカラーまたは行列で指定

meanMatrix *k* 行 2 列の平均。行列で指定

stdMatrix *k* 行 2 列の標準偏差。行列で指定

corrMatrix *k* 行 1 列の相関係数。行列で指定

colorsMatrix（オプション）k 個の等高線の色を指定する。色は、JSL で用意されている標準色の整数値、または RGB Color や HLS Color などの色関数の戻り値で指定します。

fill=x（オプション）等高線の塗りの色の透明度を指定する。

Oval(x1, y1, x2, y2, <fill>)

Oval({x1, y1}, {x2, y2}, <fill>)

説明

対角線が（*x1,y1*）と（*x2,y2*）である長方形に内接する楕円を描く。*Fill* はブール値です。*fill* が 0 の場合、楕円は塗りつぶされません。*fill* が 0 以外の場合、現在 fill color で設定されている色で塗りつぶされます。*fill* のデフォルト値は 0 です。

Path(path, <fill>)

説明

指定されたパスに沿って線を描く。fill が指定されている場合は、パスが現在の塗りつぶし色で塗りつぶされます。

引数

path N 行 3 列の行列、または SVG 構文を示す文字列

fill（オプション）線を描く（0）かパスを塗りつぶすか（1）を指定するブール値。デフォルト値は 0 です。

ノート

パスを行列で指定する場合は、x 座標、y 座標、および、パスの種類を表すフラグで構成します。フラグの値には、0（コントロール点）、1（移動）、2（線分）、3（3 次ベジエ曲線）または負の値（パスによって閉じた領域を表す場合）を指定できます。

Path To Char(path)

説明

パスの値を行列形式から文字形式に変換する。

戻り値

文字列

引数

path Nx3 のパスの行列

ノート

パスを行列で指定する場合は、x 座標、y 座標、および、パスの種類を表すフラグで構成します。フラグの値には、0（コントロール点）、1（移動）、2（線分）、3（3 次ベジエ曲線）または負の値（パスによって閉じた領域を表す場合）を指定できます。

Pen Color(n)

説明

線の色を指定する。

Pen Size(n)

説明

線の太さをピクセル値で指定する。

Pie(x1, y1, x2, y2, startangle, endangle)

説明

塗りつぶされた扇形を描く。2 つの座標で表わされる長方形に内接する楕円形が仮に構成されます。そして、開始角度（startangle）と終了角度（endangle）で指定された扇形だけが実際に描画されます。

Pixel Line To(x, y)

説明

現在の位置から、ピクセル座標で指定された位置まで、幅が 1 ピクセルの線を引く。Pixel Origin コマンドおよび Pixel Move To コマンドを使って、現在のピクセル座標を設定します。

Pixel Move To(x, y)

説明

現在の位置を、ピクセル座標で指定された新しい位置に移動する。

Pixel Origin(x, y)

説明

後に続く Pixel Line To や Pixel Move To コマンドのために、グラフ座標の原点を設定する。

Polygon({x1, y1}, {x2, y2}, ...)

Polygon(xmatrix, ymatrix)

説明

座標のリストによって定義された多角形を描く。多角形は塗りつぶされます。

Rect(x1, y1, x2, y2, <fill>)

Rect({x1, y1}, {x2, y2}, <fill>)

説明

対角線が（*x1,y1*）と（*x2,y2*）を結んだ直線である長方形を描く。*Fill* はブール値です。*fill* が 0 の場合、長方形は塗りつぶされません。*fill* が 0 以外の場合、Fill Color 関数で設定されている色で塗りつぶされます。*fill* のデフォルト値は 0 です。

RGB Color(r, g, b)

RGB Color({r, g, b})

説明

指定された赤（r）、緑（g）、および青（b）の値を JMP の色番号に変換する。

戻り値

JMP の色番号。整数

引数

赤、緑、および青、または 3 つの RGB 値を含んだリスト。指定できる値の範囲は 0 ～ 1

Text(<properties>, ({x, y}|{left, bottom, right, top}), "text")

説明

xy 座標、軸の左、下、右、一番上のいずれか、指定された位置に、引用符付き文字列（*text*）を書き込む。

プロパティ（properties）には、Center Justified（中央揃え）、Right Justified（右揃え）、Erased（消去）、Boxed（囲み）、Clockwise（時計回り）、Counterclockwise（反時計回り）のいずれかを指定できます。位置、名前付き引数、および文字列は、任意の順序で指定できます。位置と名前付き引数は、すべての文字列に適用されます。

Text Color(n)

説明

テキストの色を指定する。

Text Size(n)

説明

テキストのフォントサイズをポイント数で指定する。

Transparency(alpha)

説明

現在の描画の透明度を 0 ～ 1 の範囲の *alpha* で指定する。0 は透明（描画なし）、1 は完全に不透明（デフォルト）です。

ノート
オペレーティングシステムの中には、透化をサポートしていないものもあります。

V Line(x, <y1, y2>)

説明
x の位置に縦線を描く。*y* 座標の開始点および終了点（*y1* および *y2*）が指定された場合は、*x* の位置に *y1* から *y2* まで線を引きます。引数 *x* に行列を指定し、複数の線を描くこともできます。

V Size()

説明
グラフフレームの縦のサイズをピクセル単位で戻す。

X Function(expr, symbol, <Min(min), Max(max), Fill(value), Inc(bound), Show Details(n)>)

説明
symbol の値を Y 軸上に沿って変化させた時の関数値を X 座標にプロットする。

X Origin()

説明
グラフフレームの左端の *x* 値を戻す。

X Range()

説明
ディスプレイボックスの左端から右端までの距離を戻す。たとえば、XOrigin()+XRange() は右端を示します。

X Scale(xmin, xmax)

説明
水平方向のスケールの範囲を設定する。スケールの指定がない場合、デフォルト値の（0,100）が使われます。*xmin* のデフォルト値は 0、*xmax* のデフォルト値は 100 です。

XY Function(x(t), y(t), t, min(min), max(max), inc(bound) | steps(min))

説明
式 *x(t)* と式 *y(t)* を組み合わせ、パラメータ *t* の指定範囲で x-y 曲線を作成する。

メモ: デフォルトの精度では詳細が不明な場合は、*inc()* または *steps()* が必要です。

Y Function(expr, symbol, <Min(min), Max(max), Fill(value), Inc(bound), Show Details(n)>)

説明

symbol の値を *X* 軸上に沿って変化させた時の関数値を Y 座標にプロットする。

Y Origin()

説明

グラフフレームの下端の *y* 値を戻す。

Y Range()

説明

ディスプレイボックスの下端から上端までの距離を戻す。たとえば、YOrigin()+YRange() は上端を示します。

Y Scale(ymin, ymax)

説明

垂直方向のスケールの範囲を設定する。スケールの指定がない場合、デフォルト値の (0,100) が使われます。

リスト関数

As List(matrix)

説明

行列をリストに変換する。行列に複数の行がある場合は、行ごとのリストを含むリストを戻します。

戻り値

リスト

引数

matrix 任意の行列

Concat Items({string1, string2, ...}, <delimiter>})

説明

文字列のリストを、1 つの文字列に変換する。各文字列は、区切り文字で区切られます。区切り文字を指定しなかった場合は、スペースで区切られます。

戻り値

連結した文字列

引数

string 任意の文字列。

delimiter（オプション）各項目間に挿入される文字列。delimter には、2 文字以上指定することもできます。

例

```
str1 = "いち";
str2 = "に";
str3 = "さん";

comb = Concat Items({str1, str2, str3});
  "いち に さん"
comb = Concat Items({str1, str2, str3}, " : ");
  "いち : に : さん"
del = ",";
comb = Concat Items({str1, str2, str3}, del);
  "いち,に,さん"
```

Eval List(list)

説明

list 内の式を評価する。

戻り値

評価後の式を含むリスト

引数

リスト 有効な JSL 式のリスト

Insert(source, item, <position>)

Insert(source, key, value)

説明

ソース（*source*）の指定箇所（*position*）に新しい項目（*item*）を挿入する。*position* の指定を省略したときは、項目は最後尾に挿入されます。

連想配列の場合、ソース（*source*）の連想配列にキー（*key*）を追加し、それに値（*value*）を割り当てる。*source* にすでに *key* がある場合は、新しい *value* で置き換えられます。

引数

source 文字列、リスト、式、または連想配列。

item または key *source* の中に挿入する任意の値。連想配列の場合、*source* 内に *key* がない場合もあります。

position（オプション）*source* 内の *item* の挿入位置を示す数値。

value *key* に割り当てる値。

Insert Into(source, item, <position>)

Insert Into(source, key, value)

説明

Insert関数と同じだが、結果を元の変数に格納する。*source*は、左辺値（L-value）でなければなりません。

引数

source 文字列、リスト、ディスプレイボックス、式、または連想配列を含む変数。

item または key source の中に挿入する任意の値。連想配列の場合、*source*内に *key* がない場合もあります。

position（オプション）*source*内の *item* の挿入位置を示す数値。

value *key*に割り当てる値。

Is List(x)

説明

評価後の引数がリストのときは1、そうでなければ0を戻す。

List(a, b, c, ...)

{a, b, c, ...}

説明

一連の項目を持つリストを作成する。

N Items(source)

説明

指定されたソース（*source*）内の要素数を数える。

戻り値

リストまたはディスプレイボックスの場合は、リストまたはディスプレイボックスの中の項目の数。連想配列の場合は、キーの数。行列の場合は、行列の要素の数。名前空間の場合は、名前空間内の項目の数。

引数

source リスト、連想配列、行列、ディスプレイボックス、または名前空間。

Remove(source, position, <n>)

Remove(source, {items})

Remove(source, key)

説明

指定の場所（*position*）から数えて *n* 番目の項目を削除する。*n* が指定されていない場合、*position* にある1項目だけを削除します。*position* と *n* が指定されていない場合、最後の1項目だけを削除します。連想配列の場合、キー（*key*）およびその値を削除します。

戻り値

項目が削除されたソース（*source*）のコピー

引数

source 文字列、リスト、式、または連想配列。

position または key リストまたは式における特定の項目の位置を指す整数（または整数のリスト）。

n （オプション）削除する項目の個数を指定する整数。

Remove From(source, position, <n>)

Remove From(source, key)

説明

Remove 関数と同じだが、結果を元の変数に格納する。*n* が指定されていない場合、*position* にある 1 項目だけを削除します。*position* と *n* が指定されていない場合、最後の 1 項目だけを削除します。連想配列の場合、キー（*key*）およびその値を削除します。*source* は、左辺値（L-value）でなければなりません。

戻り値

ソース（*source*）から削除された項目のリスト

引数

source 文字列、リスト、式、ディスプレイボックス、または連想配列。

position または key リストまたは式における特定の項目の位置を指す整数（または整数のリスト）。

n （オプション）削除する項目の個数を指定する整数。

Reverse(source)

説明

ソース（*source*）の要素や項目の順序を逆にする。

引数

source 文字列、リスト、または式を含む変数。

Reverse Into(source)

説明

ソース（*source*）の要素や項目の順序を逆にし、結果を source に格納する。

引数

source 文字列、リスト、ディスプレイボックス、または式。

Shift(source, <n>)

説明

ソース（*source*）の最初の 1 つまたは *n* 個の項目を末尾に移動する。

引数

source 文字列、リスト、または式を含む変数。

n（オプション）移動する項目の個数を指定する整数。正の値を指定すると、項目をソース（*source*）の冒頭から末尾へ移動します。負の値を指定すると、項目をソース（*source*）の末尾から冒頭へ移動します。デフォルト値は1です。

Shift Into(source, <n>)

説明

Shift関数と同じだが、結果を元の変数に格納する。sourceは、左辺値（L-value）でなければなりません。

引数

source 文字列、リスト、ディスプレイボックス、または式。

n（オプション）移動する項目の個数を指定する整数。正の値を指定すると、項目をソース（*source*）の冒頭から末尾へ移動します。負の値を指定すると、項目をソース（*source*）の末尾から冒頭へ移動します。デフォルト値は1です。

Sort List(list|expr)

説明

リスト（*list*）または式（*expr*）の要素や項目を並べ替える。

Sort List Into(list|expr)

説明

Sort List関数と同じだが、結果を元の変数に格納する。リスト（*list*）や式（*expr*）はL-value（左辺に指定できるもの）でなければなりません。

Substitute(string, substring, replacementString, ...)

Substitute(list, listItem, replacementItem, ...)

Substitute(Expr(sourceExpr), Expr(findExpr), Expr(replacementExpr), ...)

説明

検索と置換を行う。第1引数で指定したソース内から、第2引数で指定した特定の部分を検索し、第3引数で指定した項目で置換します。

文字列の場合、ソース文字列（*string*）内の部分文字列（*substring*）に一致する部分を、すべて置換文字列（*replacementString*）に置換します。

リストの場合、ソースリスト（*list*）内のリスト項目（*listItem*）に一致する部分を、すべて置換項目（*replacementItem*）に置換します。

式の場合、ソース式（*sourceExpr*）内の検索式（*findExpr*）に一致する部分を、すべて置換式（*replacementExpr*）に置換します。ただし、すべての式を、Expr() 関数の中に含める必要があります。

引数

string, list, sourceExpr 置換を行う対象の文字列、リスト、式。

substring, listItem, findExpr 検索する文字列、リスト項目、式。
replacementString, replacementItem, replacementExpr 置換後の文字列、リスト項目、式。

Substitute Into(string, substring, replacementString, ...)

Substitute Into(list, listItem, replacementItem, ...)

Substitute Into(Expr(sourceExpr), Expr(findExpr), Expr(replacementExpr), ...)

説明

Substitute() と同様に検索と置換を行うが、結果を元の変数に格納する。第1引数で指定したソース内から、第2引数で指定した特定の部分を検索し、第3引数で指定した項目で置換します。第1引数は、左辺値（L-value）でなければなりません。

文字列の場合、ソース文字列（*string*）内の部分文字列（*substring*）に一致する部分を、すべて置換文字列（*replacementString*）に置換します。

リストの場合、ソースリスト（*list*）内のリスト項目（*listItem*）に一致する部分を、すべて置換項目（*replacementItem*）に置換します。

式の場合、ソース式（*sourceExpr*）内の検索式（*findExpr*）に一致する部分を、すべて置換式（*replacementExpr*）に置換します。ただし、すべての式を、Expr() 関数の中に含める必要があります。

引数

string, list, sourceExpr 置換を行う対象の文字列、リスト、式。
substring, listItem, findExpr 検索する文字列、リスト項目、式。
replacementString, replacementItem, replacementExpr 置換後の文字列、リスト項目、式。

Words("text", <"delimiters">)

説明

引数の文字列（*text*）から単語を抽出する。区切り文字（delimiters）を指定すると、文字列を各単語に区切る際に、その文字が使われます。デフォルトの区切り文字はスペースです。複数の区切り文字を指定した場合、指定したすべての文字が区切り文字とみなされます。

例

```
Words( "the quick brown fox" );
  {"the","quick","brown","fox"}
Words( "Doe, Jane P.",", ." );
  {"Doe","Jane","P"}
```

MATLABインテグレーション関数

JMPには、MATLABへのインターフェースが関数として用意されています。基本的には、まず、MATLABへの接続を開始し、次に、何らかの処理をMATLAB上で実行し、最後にMATLABへの接続を解除します。これらの関数の多くは、MATLABの処理が正常に実行された場合は0、そうでない場合は、エラーコードを戻します。MATLABの処理が正常に実行されなかった場合は、「ログ」ウィンドウにメッセージが表示されます。ただし、MATLAB Get( )関数だけは、エラーコード以外の値を戻します。

MATLABの使用方法については、『スクリプトガイド』の「JMPの拡張」章を参照してください。

MATLAB JSL関数インターフェース

MATLAB Connect(<named arguments>)

説明
現在の MATLAB 接続オブジェクトを戻します。もし、現在、MATLAB に接続されていない場合は、JMP から MATLAB への接続を初期化した後、MATLAB 接続オブジェクトを戻します。

戻り値
スクリプト可能な MATLAB オブジェクト

名前付き引数
Echo(Boolean) 実行したMATLABのプログラムコードを、JMPのログに出力する。デフォルト値は1（真）。

MATLAB Control(<named arguments>)

説明
プログラムコードの表示などの外部イベントの制御命令を MATLAB に送る。

戻り値
なし

引数
なし

名前付き引数
Echo(Boolean) グローバル。実行した MATLAB のプログラムコードを、JMP のログに出力する。
Visible(Boolean) グローバル。アクティブな MATLAB ワークスペースの表示／非表示を指定する。

MATLAB Execute({ list of inputs }, { list of outputs }, mCode, <named arguments>)

説明
現在のグローバルな MATLAB 接続に対して、第 1 引数のリストに指定された JMP 変数を送り、第 3 引数に指定された MATLAB コードをサブミットする。第 2 引数のリストに指定された変数が、JMP に戻されます。

戻り値

成功した場合は0、そうでない場合は0以外。

引数

`{ list of inputs }` 位置固定、名前のリスト。入力としてMATLABに送られるJMP変数名のリスト。

`{ list of outputs }` 位置固定、名前のリスト。出力としてMATLABから取得されるJMP変数名のリスト。

`mCode` 位置固定、文字列。サブミットされるMATLABコード。

名前付き引数

`Expand(Boolean)` MATLABコードのサブミット前に、コードに対してEval Insertを実行する。

`Echo(Boolean)` 実行したMATLABのプログラムコードを、JMPのログに出力する。デフォルト値は1（真）です。

例

次の例では、JMP変数xとyをMATLABに送り、MATLABステートメント *z = x * y* を実行して、MATLAB変数zを取得してJMPに戻します。

```
MATLAB Init();
x = [1 2 3];
y = [4 5 6];
MATLAB Execute( {x, y}, {z}, "z = x * y;" );
Show( z );
```

MATLAB Get(name)

説明

name引数で指定されたMATLABの変数を、JMPで取得する。

戻り値

name引数で指定された変数の値。

引数

`name` 位置固定。MATLABに送るJMP変数の名前。

例

qbxという名前の行列と、dfという名前の構造体が、MATLAB上に存在するとします。

```
// MATLAB 変数 qbx の値を取得し、それを JMP 変数 qbx に代入
qbx = MATLAB Get( qbx );

// MATLAB 変数 df のデータフレームを、JMP の変数 df にて参照される JMP データテーブルとして取得
df = MATLAB Get( df );
```

表2.1に、MATLAB Get( )関数でMATLABからJMPに変数を取得した場合に、MATLABのデータタイプ（データ型）が、JMPにおいて、どのような型に変換されるかを示します。リストの場合は、リスト内の要素ごとにデータタイプをチェックして、変換します。入れ子になったリストもサポートされます。

表2.1 MATLAB Get( )関数のJMPデータタイプとMATLABデータタイプ

MATLABデータタイプ	JMPデータタイプ
実数（double）	数値
論理	数値 (0 | 1)
文字列	文字列
整数	数値
日付／時間	数値
構造体	データテーブル
行列	行列
数値ベクトル	行列
文字列ベクトル	文字列のリスト
グラフ	ピクチャーオブジェクト

MATLAB Get Graphics(format)

説明

MATLAB グラフ表示ウィンドウに書き込まれた最後のグラフオブジェクトを、format 引数で指定されたグラフィック形式で取得する。

戻り値

JMP ピクチャーオブジェクト。

引数

format 位置固定。MATLAB のグラフを取得するときの変換形式。有効な形式は、"png"、"bmp"、"jpeg"、"jpg"、"tiff"、および "tif" です。

MATLAB Init(<named arguments>)

説明

MATLAB 接続インターフェースを初期化する。

戻り値

リターンコード。

名前付き引数

Echo(Boolean) は、実行した MATLAB のプログラムコードを、JMP のログに出力する。このオプションは、グローバルです。デフォルト値は 1（真）。

MATLAB Is Connected()

説明

MATLAB 接続がアクティブかどうかを調べる。

戻り値

アクティブな MATLAB 接続がある場合は 1、そうでない場合は 0 を戻す。

MATLAB JMP Name To MATLAB Name(name)

説明

MATLAB の命名規則に従い、JMP 変数名を、対応する MATLAB 変数名に変換する。

戻り値

マップされた MATLAB の変数名を文字列として戻す。

引数

`name` 位置固定。MATLAB に送る JMP 変数の名前。

MATLAB Send(name, <named arguments>)

説明

名前付き変数を JMP から MATLAB に送る。

戻り値

成功した場合は 0、そうでない場合は 0 以外。

引数

`name` 位置固定。MATLAB に送る JMP 変数の名前。

名前付き引数

次のオプションの引数は、データテーブルにのみ適用されます。

`Selected(Boolean)` 参照先データテーブルの選択されている行を MATLAB に送る。

`Excluded(Boolean)` 参照先データテーブルの除外されている行のみを MATLAB に送る。

`Labeled(Boolean)` 参照先データテーブルのラベルのついている行のみを MATLAB に送る。

`Hidden(Boolean)` 参照先データテーブルの非表示の行のみを MATLAB に送る。

`Colored(Boolean)` 参照先データテーブルの色のついている行のみを MATLAB に送る。

`Markered(Boolean)` 参照先データテーブルのマーカーのついている行のみを MATLAB に送る。

`Row States(Boolean, <named arguments>)` 参照先データテーブルにおける行属性の情報を、「RowState」という列名のデータ列を追加して、MATLAB に送る。個々の設定をあわせて追加すれば、複数選択も可能です。行属性の各設定には、それぞれ次の値が割り当てられています。

- Selected = 1
- Excluded = 2
- Labeled = 4
- Hidden = 8

– Colored = 16

– Markered = 32

Row States には、次の名前付き引数をオプションで指定できます。

`Colors(Boolean)` 行の色を送る。「RowStateColor」という名前のデータ列を追加します。

`Markers(Boolean)` 行のマーカーを送る。「RowStateMarker」という名前のデータ列を追加します。

例

```
// 行列を変数 X に代入し、その変数を MATLAB に送る。
X = [1 2 3];
ml = MATLAB Send( X );

// データテーブルを開き、その参照を dt に割り当てる。そして、そのデータテーブルを、現在の行属性も
   含めて MATLAB に送る。
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
ml = MATLAB Send( dt, Row States( 1 ) );
```

表2.2に、MATLAB Send( )関数でJMPからMATLABに変数を送った場合に、JMPのデータタイプ（データ型）が、MATLABにおいて、どのような型に変換されるかを示します。リストの場合は、リスト内の要素ごとにデータタイプをチェックして、変換します。入れ子になったリストもサポートされます。

表2.2 MATLAB Send( )関数のJMPデータタイプとMATLABデータタイプ

MATLABデータタイプ	JMPデータタイプ
実数（double）	数値
文字列	文字列
実数の行列	行列
構造体	データテーブル

例

```
MATLAB Init( );
X = 1;
MATLAB Send( X );
S = " レポートのタイトル ";
MATLAB Send( S );
M = [1 2 3, 4 5 6, 7 8 9];
MATLAB Send( M );
MATLAB Submit( "
X
S
M
" );
MATLAB Term( );
```

MATLAB Submit File('pathname', <named arguments>)

説明

pathname で指定されたファイルに含まれる MATLAB コードを、MATLAB 上で実行する。

戻り値

成功した場合は 0、そうでない場合は 0 以外。

引数

pathname 位置固定、文字列。実行する MATLAB プログラムコードを含むファイルのパス名。

名前付き引数

Expand(Boolean) MATLAB コードのサブミット前に、コードに対して Eval Insert を実行する。

Echo(Boolean) 実行した MATLAB のプログラムコードを、JMP のログに出力する。デフォルト値は 1（真）です。

MATLAB Submit(mCode, <named arguments>)

説明

アクティブなグローバル MATLAB 接続に、MATLAB コードをサブミットする。

戻り値

成功した場合は 0、そうでない場合は 0 以外。

引数

mCode 位置固定、文字列。サブミットされる MATLAB コード。

名前付き引数

Expand(Boolean) MATLAB コードのサブミット前に、コードに対して Eval Insert を実行する。

Echo(Boolean) 実行した MATLAB のプログラムコードを、JMP のログに出力する。デフォルト値は 1（真）です。

例

次の例は、2 つの乱数ベクトルを作成し、それらを x 変数と y 変数にプロットします。

```
MATLAB Init();
mc = MATLAB Submit("/[
   x = rand(5);
   fprintf('%f/n', x);
   y = rand(5);
   fprintf('%f/n', x);
   z = plot(x, y);
]/" );
```

MATLAB Term();

説明

MATLAB への接続を終了する（アクティブな MATLAB 接続インターフェースを終了する）。

戻り値
アクティブな MATLAB 接続がある場合は 1、そうでない場合は 0

引数
なし

行列関数

All(A, ...)

戻り値
すべての行列のすべての要素が 0 以外のときは 1、そうでなければ 0

Any(A, ...)

戻り値
1 つまたは複数の行列のある 1 要素が 0 以外のときは 1、そうでなければ 0

CDF(Y)

説明
ベクトルまたはリストで指定された *Y* に対し、経験累積分布関数の値を戻す。経験累積分布関数は、*QuantVec* の値以下となっているデータの割合を表します。

構文
```
{QuantVec, CumProbVec} = CDF(YVec)
```

Chol Update(L, V, C)

説明
n×*n* 行列 A のコレスキー根を L とした場合、cholUpdate を呼び出すと、A+V*C*V'（C は *m*×*m* の対称行列、V は n×m 行列）のコレスキー根で置き換わる。

Cholesky(A)

説明
半正値定符号行列の下コレスキー根（L）を戻す。L は、L*L' = A となるような下三角行列。

戻り値
L（コレスキー根）

引数
A 対称行列

Correlation(matrix)

説明

行列引数 *matrix* の相関係数行列を計算する。

戻り値

指定された行列の相関係数行列

引数

matrix データを示す行列。データが *m* 行 *n* 列で構成されている場合、結果は m × m の行列です。

ノート

欠測値が 1 つでもある行は、行ごと除外されます。

この関数は、可能ならマルチスレッドを使用します。そのため、行数が多い大きなデータに適しています。

列が定数の場合、その相関係数は 0 で、対角要素も 0 です。

Covariance(matrix, < <<"Pairwise">, < <<"Shrink">, < <<Freq(vector)>, < <<Weight(vector)>)

説明

行列引数 *matrix* の共分散行列を計算する。

戻り値

指定された行列の共分散行列

引数

matrix データを示す行列。データが *m* 行 *n* 列で構成されている場合、結果は m × n の行列です。

"Pairwise" 欠測値に対してリストワイズ法ではなくペアワイズ法を使用する。

"Shrink" Schafer-Strimmer の縮小推定を実行する。

<<Freq(vector) 行列の行の度数を指定するベクトル。

<<Weight(vector) 行列の行の重みを指定するベクトル。

ノート

デフォルトでは、欠測値が 1 つでもある行は、行ごと除外されます。"Pairwise" オプションを指定した場合は、欠測値以外の全ペアを共分散行列の計算に使用します。

この関数は、可能ならマルチスレッドを使用します。そのため、行数が多い大きなデータに適しています。

Design(vector, < levelsList | <<levels, <<ElseMissing >)

説明

ベクトル（*vector*）の一意の値ごとに、1 と 0 の列でできた計画行列を作成する。

戻り値

引数の一意の値ごとに 1 と 0 の列を持つ計画行列、または計画行列と水準のリストを含むリスト。

引数

vector ベクトル

levelsList（オプション）戻される行列の水準を指定するリストまたは行列。
<<levels（オプション）これを指定すると、戻り値が計画行列と水準を含むリストになる。
<<ElseMissing（オプション）引数 *vector* に、引数 *levelsList* に含まれない値があった場合、その値の処理を指定する。この引数を指定すると、計画行列に欠測値が挿入されます。指定しない場合は、計画行列に 0 が挿入されます。

ノート

*levelsList*引数内の欠測値は無視されません。たとえば、次のように実行します。

```
Show( Design ( ., [. 0 1] ),
Design( 0, [. 0 1] ),
Design( 1, [. 0 1] ),
Design( [0 0 1 . 1], [. 0 1] ),
Design( {0, 0, 1, ., 1}, [. 0 1] ) );
  Design(., [. 0 1]) = [1 0 0];
  Design(0, [. 0 1]) = [0 1 0];
  Design(1, [. 0 1]) = [0 0 1];
  Design([0 0 1 . 1], [. 0 1]) =
  [ 0 1 0,
    0 1 0,
    0 0 1,
    1 0 0,
    0 0 1];
  Design({0, 0, 1, ., 1}, [. 0 1]) =
  [ 0 1 0,
    0 1 0,
    0 0 1,
    1 0 0,
    0 0 1];
```

Design Nom(vector, < levelsList | <<levels, <<ElseMissing >)

DesignF(vector, < levelsList | <<levels, <<ElseMissing >)

説明

引数の一意の値ごとに、1 と 0 の列を持つ計画行列を作成する。ただし、最後の水準は、-1 の行としてコード化されます。

戻り値

フルランクの計画行列、または計画行列と水準を含むリスト

引数

vector ベクトル
levelsList（オプション）戻される行列の水準を指定するリストまたは行列。この引数を指定すると、このリストまたは行列の最後の水準が、計画行列の最後の水準として扱われます。指定しない場合は、引数 *vector* の最大値が、最後の水準に設定されます。
<<levels（オプション）これを指定すると、戻り値が計画行列と水準を含むリストになる。

<<ElseMissing（オプション）引数 *vector* に、引数 *levelsList* に含まれない値があった場合、その値の処理を指定する。この引数を指定すると、計画行列に欠測値が挿入されます。指定しない場合は、計画行列に 0 が挿入されます。

ノート

levelsList 引数内の欠測値は無視されません。たとえば、次のように実行します。

```
Show( Design Nom( ., [. 0 1] ),
Design Nom( 0, [. 0 1] ),
Design Nom( 1, [. 0 1] ),
Design Nom( [0 0 1 . 1], [. 0 1] ),
Design Nom( {0, 0, 1, ., 1}, [. 0 1] ) );
  Design Nom(., [. 0 1]) = [1 0];
  Design Nom(0, [. 0 1]) = [0 1];
  Design Nom(1, [. 0 1]) = [-1 -1];
  Design Nom([0 0 1 . 1], [. 0 1]) = [0 1, 0 1, -1 -1, 1 0, -1 -1];
  Design Nom({0, 0, 1, ., 1}, [. 0 1]) = [0 1, 0 1, -1 -1, 1 0, -1 -1];
```

Design Ord(vector, < levelsList | <<levels, <<ElseMissing >)

説明

引数の一意の値ごとに列を持つ計画行列を作成する。最初の水準は、0 の行としてコード化されます。引数 *levelsList* のそれ以降の n 番目の水準は、(n-1) 個の 1 およびそれ以外が 0 の行としてコード化されます。

戻り値

フルランクの計画行列、または計画行列と水準を含むリスト

引数

vector ベクトル

levelsList（オプション）戻される行列の水準を指定するリストまたは行列。

<<levels（オプション）これを指定すると、戻り値が計画行列と水準を含むリストになる。

<<ElseMissing（オプション）引数 *vector* に、引数 *levelsList* に含まれない値があった場合、その値の処理を指定する。この引数を指定すると、計画行列に欠測値が挿入されます。指定しない場合は、計画行列に 0 が挿入されます。

ノート

levelsList 引数内の欠測値は無視されません。たとえば、次のように実行します。

```
Show( Design Ord( ., [. 0 1] ),
Design Ord( 0, [. 0 1] ),
Design Ord( 1, [. 0 1] ),
Design Ord( [0 0 1 . 1], [. 0 1] ),
Design Ord( {0, 0, 1, ., 1}, [. 0 1] ) );
  Design Ord(., [. 0 1]) = [0 0];
  Design Ord(0, [. 0 1]) = [1 0];
  Design Ord(1, [. 0 1]) = [1 1];
  Design Ord([0 0 1 . 1], [. 0 1]) = [1 0, 1 0, 1 1, 0 0, 1 1];
  Design Ord({0, 0, 1, ., 1}, [. 0 1]) = [1 0, 1 0, 1 1, 0 0, 1 1];
```

Det(A)

説明
正方行列の行列式。

戻り値
行列式

引数
A 正方行列

Diag(A, <B>)

説明
正方行列やベクトルから対角行列を作成する。2つ以上の行列が指定された場合、それらの行列を対角線上に連結したブロック対角行列を戻します。

戻り値
行列

引数
A 行列またはベクトル

Direct Product(A, B)

説明
行列の直積（Kronecker 積）を戻す。

戻り値
n の階乗

引数
A, B 行列

Distance(x1, x2, <scales>, <powers>)

説明
x1 の行と x2 の行との間の距離の行列を生成する。

戻り値
行列

引数
x1, x2 2つの行列。
scales（オプション）行列の尺度を設定する引数
powers（オプション）行列のべき乗を設定する引数

E Div(A, B)

A/B

説明
2つの行列の要素ごとの割り算。

戻り値
結果の行列

引数
A, B 2つの行列。

E Mult(A, B)

A*B

説明
2つの行列の要素ごとの掛け算。

戻り値
結果の行列

引数
A, B 2つの行列。

Eigen(A)

説明
固有値分解。

戻り値
対称行列Aに対して、A=E * Diag(M) * E`となるようなMとEが、リスト {M,E} の形式で戻される。

引数
A 対称行列

G Inverse(A)

説明
Moore-Penrose型の一般逆行列。

H Direct Product(A, B)

説明
行数が等しい行列の横の直積。

Hough Line Transform(matrix, <NAngle(number)>, <NRadius(number)>)

説明

彩度の行列をとり、それを行列内の線を見つけやすいように変換する。角度を列、半径を行とした Hough Line Transform を含む行列を作成します。

引数

matrix イメージの彩度から得られた行列の場合もあるが、平坦化の装置によって筋状に問題が発生している可能性のある半導体ウエハーから得られたものである場合が多い。

NAngle(number) 異なるサイズで変換するための角度を入力する。デフォルト値は 180 度です。

NRadius(number) 異なるサイズで変換するための半径を入力する。デフォルトは sqrt(NRow*nRow+nCol*nCol) です。

Identity(n)

説明

n × *n* の単位行列を作成する。単位行列は、対角要素が 1 で、非対角要素が 0 である行列です。

戻り値

行列

引数

n 整数

Index(i, j, <increment>)

i::j

説明

i から *j* までの整数を含む行ベクトルを作成する。

戻り値

行列

引数

i, j 範囲を定義する整数。*i* は範囲の始点、*j* は終点

increment（オプション）これを指定することにより、増分をデフォルト値の +1 から変更できる

Inv()

「Inverse(A)」（145ページ）を参照してください。

Inv Update(A, x, 1|-1)

説明

A=Inv(**X`X**) がすでに計算されている場合、行列 X に行ベクトル x を追加／削除した後の、Inv(X`X) を効率的に計算する。

引数

A 更新する行列

X 行列 *A* に対して追加／削除する 1 つまたは複数の行

1|-1 第 3 引数は、第 2 引数 *x* で指定された行を行列 *A* に追加／削除するかどうかを制御する。1 は行を追加し、-1 は行を削除します。

Inverse(A)

Inv(A)

説明

逆行列を戻す。行列は正則な正方行列でなければなりません。

Is Matrix(x)

説明

評価後の引数が行列のときは 1、そうでなければ 0 を戻す。

J(nrows, <ncols>, <value>)

説明

すべての要素が同じ値（value）の行列を作成する。

戻り値

行列

引数

nrows 行列の行数。列数（ncols）が指定されていない場合、列数は行数と同じ数に設定されます。

ncols 行列の列数

value 行列を埋める値。value が指定されていない場合、1 が使用されます。

KDTable(matrix)

説明

近傍点を効率よく検索するためのテーブルを戻す。

戻り値

K 次元テーブルオブジェクト

引数

matrix K 次元上における点の座標を表す行列。次元数または点数に制限はありません。行列の各列はデータの次元、各行はデータ点を表しています。

メッセージ

<<Distance between rows(row1, row2) K 次元テーブル内の指定された 2 つの行の間の距離を戻す。この距離は、削除された行および挿入された行に対しても適用されます。

<<K nearest rows(stop, <position>) n 個の近傍点と、それまでの距離を含む行列を、結果として戻す。*position* が指定されていない場合は、すべての行に関する結果を戻します。*position* が

指定されている場合は、それで指定された点に対する結果を戻します。引数 *stop* には、*n* または {*n, limit*} を指定してください。引数 position には、n 個の近傍点を求めたい点を、座標の行ベクトル、もしくは、行番号の整数で指定してください。

<<Remove rows(number | vector) *number* で指定された行、または、*vector* で指定された行を、削除する。削除された行の行数を戻します。すでに削除されている行は無視されます。

<<Insert rows(number | vector) *number* で指定された行、または、*vector* で指定された行を、再度挿入する。挿入された行の行数を戻します。すでに挿入されている行は無視されます。

ノート

行が削除または挿入されても、行のインデックスは変更されません。削除または再挿入できるのは、既存の次元データテーブルオブジェクトに含まれている行だけです。新しい行を含める必要がある場合は、別の次元データテーブルを新たに作成してください。

Least Squares Solve(X, y, messages)

説明

最小 2 乗法を実行する（オプションで重み付きも可能）。

戻り値

行列 Beta=Inverse(X'X')X'y および Beta の分散共分散行列の推定値を含むリスト。

引数

<<weights(optional weight vector) 重み（ベクトル）を指定して、重み付き最小 2 乗法を実行する。

<<method("Sweep" | "GInv") 計算効率が良い Sweep 法（デフォルト）か、数値的に安定する一般逆行列法（GInv）のどちらで正規方程式を解くかを指定する。

Loc(A)

Loc(list, item)

説明

行列 *A* の要素のうち、0 でも欠測値でもない要素の位置を示す、添え字の行列を戻す。引数が 2 つ指定された場合は、リスト *list* のうち要素 *item* が見つかった位置の行列を戻します。

戻り値

行列

引数

A 行列

リスト リスト

item リスト内で探す要素

Loc Max(A)

説明

行列の要素のうち、最小値である要素の位置を戻す。

戻り値
最小値の位置を示す整数

引数
A 行列

Loc Min(A)

説明
行列の要素のうち、最小値である要素の位置を戻す。

戻り値
最小値の位置を示す整数

引数
A 行列

Loc NonMissing(matrix, ..., {list}, ...)

説明
指定された行列またはリスト内で、欠測値がない行の番号を戻す。引数がリストの場合、空でない文字列の番号も戻すことができます。

戻り値
新しい行列またはリスト

Loc Sorted(A, B)

説明
行列 *B* の各要素について、その値以下の最大値を行列 *A* から検索し、その位置を行列で戻す。*A* は昇順に並べた行列でなければなりません。

戻り値
行列

引数
A, B 行列

Matrix({{x11, x12, ..., x1m}, {x21, x22, ..., 2m}, {...}, {xn1, xn2, ..., xnm}})

説明
m 個の値を行とする *n* 個のリストから、または指定した行と列の数から、*m*×*n* の行列を作成する。

戻り値
行列

引数
指定の値で構成されたリストを行とするリスト

例

```
mymatrix = Matrix({{1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {10, 11, 12}});
  [ 1 2 3,
    4 5 6,
    7 8 9,
    10 11 12]
```

等価表現

```
[x11 x12 ... x1m,
   ...,
   xn1 xn2 ... xnm ]
```

Matrix Mult(A, B)

C=A*B, ...

説明

行列の掛け算。

引数

行数と列数が整合している 2 つ以上の行列（すべての乗算において、掛けられる行列の列数が掛ける行列の行数と一致している必要がある）。

ノート

Matrix Mult() で指定できる引数は 2 つだけですが、* 演算子を使用すれば 3 つ以上の行列を掛けることができます。

Mode(list or matrix)

説明

数値または文字のリスト、あるいは数値行列から、最も頻繁に出現する項目を選択する。頻度の同じものがある場合は、最も小さい値が選択されます。複数の引数を指定する場合は、数値と文字列を組み合わせることもできます。

引数

リストまたは行列を指定する。

Multivariate Normal Impute(yVec, meanYvec, symCovMat)

説明

応答ベクトル yVec における欠測値を平均と共分散に基づいて補完する。

引数

yVec いくつかの要素が欠測値となっているベクトル

meanYvec 平均のベクトル

symCovMat 共分散行列（対称行列）。共分散行列を指定しなかった場合は、平均で補完されます。

NChooseK Matrix(n, k)

説明

n 個から *k* 個を選んだ場合のすべての組み合わせを含む行列を戻す。

N Col(x)

N Cols(x)

説明

指定されたデータテーブルまたは行列の列数を戻す。

引数

x データテーブルまたは行列

Ortho(A, <Centered(0)>, <Scaled(1)>)

説明

行列 *A* の列を Gram-Schmidt 法で直交正規化する。Centered(0) を指定すると、直交化後の列の和がゼロに中心化されない。Scaled(0) を指定すると、長さが 1 に尺度化されない。

Ortho Poly(vector, order)

説明

説明変数の間隔を表すベクトル（*vector*）に対し、指定の次数（*order*）までの直交多項式を戻す。

Print Matrix(M, <named arguments>)

説明

表示形式を整えた行列を含む文字列を戻す。たとえば、この関数を使用して、行列をログに出力できます。

引数

M 行列

名前付き引数

以下の名前付き引数は、すべてオプションです。

<<ignore locale(Boolean) 偽（0）の場合は、ロケールの小数点記号が使われる。真（1）の場合は、常にピリオド（.）を小数点記号とする。デフォルト値は 0（偽）です。

<<decimal digits(n) 表示する小数桁数を指定する整数。

<<style("style name") Parseable、Latex、Other のいずれかのスタイルを指定する。Parseable は JSL と同じ表示形式です。Latex は LaTex 用の形式です。Other を指定した場合、次の 3 つの引数を指定する必要があります。

<<separate("character") 要素の区切り記号を定義する。

<<line begin("character") 開始行の文字を定義する。

<<line end("character") 終了行の文字を定義する。

QR(A)

説明

A の QR 分解を戻す。通常は、{Q, R} = QR(A) のように使います。

Rank Index(vector)

Rank(vector)

説明

引数のベクトルを昇順に並べ替えるためのインデックスを戻す。結果のベクトルを、元のベクトル（*vector*）の添え字として使用すれば昇順に並べ替えることができる。なお、欠測値は結果から除外されます。行列に加えて、数値または文字列のリストがサポートされています。

Ranking(vector)

説明

ベクトル（*vector*）の各要素の大きさの順位を、列ベクトルで戻す。1 ～ *n* の順位が低位から高位に与えられますが、同じ値に対しては任意の順位が与えられます。行列に加えて、数値または文字列のリストがサポートされています。

Ranking Tie(vector)

説明

ベクトル（*vector*）の各要素の大きさの順位を、列ベクトルで戻す。ただし、同じ値に対しては、平均順位が与えられます。行列に加えて、数値または文字列のリストがサポートされています。

Shape(A, nrow, <ncol>)

説明

行列 *A* の全行を指定の次元に再構成する。行列 A の各値は、新しい行列に行ごとに配置されます。

戻り値

再構成された行列

引数

A 行列
nrow 新しい行列の行数を指定
ncol（オプション）新しい行列の列数を指定

ノート

列数（*ncol*）が指定されていない場合は、元の行列の値をすべて挿入できるだけの列数が使用されます。

新しい行列が元の行列よりも小さい場合は、余った値が破棄されます。

新しい行列が元の行列よりも大きい場合は、新しい行列が完全に埋まるまで値が繰り返されます。

例

```
a = Matrix({ {1, 2, 3}, {4, 5, 6}, {7, 8, 9} });
  [ 1 2 3,
```

```
    4 5 6,
    7 8 9]

Shape(a, 2);
  [ 1 2 3 4 5,
    6 7 8 9 1]

Shape(a, 2, 2);
  [ 1 2,
    3 4]

Shape(a, 4, 4);
  [ 1 2 3 4,
    5 6 7 8,
    9 1 2 3,
    4 5 6 7]
```

Solve(A, b)

説明

線形システムの解を戻す。この解は、x=inverse(A)*b によって得られる解と同じです。

Sort Ascending(source)

説明

リスト（*source*）の要素を昇順に並べたリストを戻す。

Sort Descending(source)

説明

リスト（*source*）の要素を降順に並べたリストを戻す。

Spline Coef(x, y, lambda)

説明

knots||a||b||c||d 形式の 5 個の要素が含まれた列ベクトルを戻す。ここで、knots は節点を表し、*x* の一意な値で構成されています。

x は説明変数のベクトル、y は応答変数のベクトル、lambda は平滑化パラメータです。lambda の値が大きくなるほどなめらかなスプライン曲線になります。

Spline Eval(x, coef)

説明

Spline Eval 関数は、Spline Coef 関数から戻される係数の行列（これは、3 次スプラインの場合、knots||a||b||c||d という形式の行列です）から、スプライン曲線の予測値を計算します。説明変数を表す引数の x は、スカラーでも行列でもかまいません。引数の coef 行列には、2 列以上の任意の列数を

設定することができ、各列がそれぞれの次数の係数を表します。x の累乗を計算するときには、節点（knots）の値が引かれます。たとえば、引数 coef が knots||a||b||c||d である場合、x よりも小さいもののなかで最も大きな節点を knots[j] と表し、xx = x - knots[j] とすると、*x* に対する予測値は次のように計算されます。

```
result = a[j] + xx * (b[j] + xx * (c[j] + xx * d[j]))
```

変形すると、次のようになります。

```
result = a[j] + b[j] * xx + c[j] * xx ^ 2 + d[j] * xx ^ 3
```

Spline Smooth(x, y, lambda)

説明

スプライン曲線をあてはめた結果の予測値 (平滑化された値) を戻す。

x は説明変数のベクトル、y は応答変数のベクトル、lambda は平滑化パラメータです。lambda の値が大きくなるほどなめらかなスプライン曲線になります。

SVD(A)

説明

特異値分解。

Sweep(A, <indices>)

説明

掃き出し法による行列の計算を行う。

Trace(A)

説明

トレース、つまり正方行列の対角要素の和。

Transpose(A)

説明

行列 *A* の行と列を転置する。

戻り値

転置した行列

引数

A 行列

等価表現

A`

V Concat(A, B, ...)

説明
2つ以上の行列を縦に連結する。

戻り値
行列

引数
2つ以上の行列

V Concat To(A, B, ...)

説明
2つの行列を縦に連結し、その結果を元の変数に割り当てる。

戻り値
行列

引数
2つ以上の行列

V Max(matrix)

説明
行列（*matrix*）の各列の最大値を含む行ベクトルを戻す。

V Mean(matrix)

説明
行列（*matrix*）の各列の平均を含む行ベクトルを戻す。

V Min(matrix)

説明
行列（*matrix*）の各列の最小値を含む行ベクトルを戻す。

V Standardize(matrix)

説明
行列の各列を、平均0、標準偏差1に標準化して戻す。

V Std(matrix)

説明
行列（*matrix*）の各列の標準偏差を含む行ベクトルを戻す。

V Sum(matrix)

説明
行列（*matrix*）の各列の合計を含む行ベクトルを戻す。

Varimax(matrix)

説明
回転後の行列と直行回転に用いた行列をリストで戻す。

Vec Diag(A)

説明
正方行列 *A* の対角要素から成るベクトルを作成する。

戻り値
行列

引数
正方行列

ノート
正方行列でない行列を指定した場合はエラーが生じます。

Vec Quadratic(symmetric matrix, rectangular matrix)

説明
n × *1* 行列を作成する。ハット値などの計算に使われます。

戻り値
行列

引数
2 つの行列。最初の行列は対称行列でなければならない

等価表現
```
Vec Diag(X*Sym*X')
```

数値関数

Abs(n)

説明
n の絶対値を計算する。

戻り値
n の絶対値

引数

n 任意の数値

Ceiling(n)

説明

n が整数でない場合、*n* を整数に切り上げた値を戻す。

戻り値

n 以上の整数のうち最小のもの

引数

n 任意の数値

Derivative(expr, {name, ...}, ...)

説明

name に対する *expr* 式の導関数を計算する。

戻り値

導関数

引数

expr 任意の式 この引数は、Name Expr、Expr、Eval などの関数を用いて、間接的に指定してもかまいません。

name 1つの変数または変数のリスト

ノート

追加の変数をリスト内に指定すると（Derivative(expr, {name}, {name2})）、第2次導関数も求められます。

Floor(n)

説明

n が整数でない場合、*n* を整数に切り捨てた値を戻す。

戻り値

n 以下の整数のうち最大のもの

引数

n 任意の数値

例

```
Floor( 2.7 );
  2
Floor( -.5 );
  -1
```

Invert Expr(expr, name)

説明

名前（*name*）に関して、式（*expr*）の逆関数を求めることを試みる。

Mod()

「Modulo(number, divisor)」（156ページ）を参照

Modulo(number, divisor)

Mod(number, divisor)

説明

数（*number*）を除数（*divisor*）で割った余りを戻す。

例

```
Modulo( 6, 5 );
  1
```

Normal Integrate(muVector, sigmaMatrix, expr, x, nStrata, nSim)

説明

多変量の正規分布に従う変数の滑らかな関数を動径 - 球法で積分した結果を戻す。

引数

muVector ベクトル
sigmaMatrix 行列
expr 変数 *x* で表される式
x 式 *expr* に使用される変数
nStrata 層の数
nSim シミュレーションの回数

Num Deriv(f(x,...), <parnum=1>)

説明

関数 *f(x,...)* を数値微分により偏微分した結果を戻す。*Num Deriv* 関数の第2引数に、偏微分する引数を指定する。第2引数を指定しなかった場合、関数の最初の引数に対する偏微分が求められる。偏微分は *f(x,...)* の引数に指定されている数値の箇所で評価される。

ノート

関数 Num Deriv() の結果は、以下の例のように、間違っているように見えることがあります。

```
x = 3;
n = Num Deriv( 3 * x ^ 2 );
    // 9.00000000001455
```

この使用法は正しくありません。この関数は、「非線形回帰」プラットフォームで、解析的な微分がわからない関数を微分するために使用されることを想定しています。この関数の正しい使い方は以下のとおりです。

```
x = 3;
f = Function( {x}, 3 * x ^ 2 );
n = Num Deriv( f( x ), 1 );
  // 18.000029999854
```

Num Deriv2(f(x,...))

説明

関数 *f(x,...)* を *x* に関して数値微分により 2 次微分した結果を戻す。偏微分は *f(x,...)* の引数に指定されている数値の箇所で評価される。

Round(n, places)

説明

n を四捨五入して、桁数（*places*）で指定された小数点以下の桁数に丸める。

Simplify Expr(expr(expression))

Simplify Expr(nameExpr(global))

説明

式を代数的に簡素化する。

最適化関数

Constrained Maximize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

説明

式 *expr* を最大化する引数 *x* の値を求める。引数はリストで指定すること。引数名の後に括弧で囲んだ数値を指定することにより、各引数の下限と上限を指定できる。

以下のメッセージにおいて、A は係数からなる行列、x = [x1, x2, ...] は引数のベクトル、b は、式の右辺を成すベクトルとする。

メッセージ

<<Less than EQ({A, b}) 指定された値以下に制約する (A*x <= b)。

<<Greater Than EQ({A, b}) 指定された値以上に制約する (A*x >= b)。

<<Equal To({A, b}) 指定された値に制約する (A*x = b)。

<<Starting Values([x1Start, x2Start, ...]) 初期値を指定する。

<<Max Iter(int) 実行される反復の最大回数を指定する整数。

<<Tolerance(p) *p* は収束基準値。デフォルト値は 10^5。

<<Show Details("true") 結果のリスト（目標値、反復回数、勾配、ヘッセ行列）を戻す。最適化のステップごとの結果をログに出力する。

Constrained Minimize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

説明

式 *expr* を最小化する引数 *x* の値を求める。引数はリストで指定すること。引数名の後に括弧で囲んだ数値を指定することにより、各引数の下限と上限を指定できる。

以下のメッセージにおいて、A は係数からなる行列、x = [x1, x2, ...] は引数のベクトル、b は、式の右辺を成すベクトルとする。

メッセージ

<<Less than EQ({A, b}) 指定された値以下に制約する (A*x <= b)。

<<Greater Than EQ({A, b}) 指定された値以上に制約する (A*x >= b)。

<<Equal To({A, b}) 指定された値に制約する (A*x = b)。

<<Starting Values([x1Start, x2Start, ...]) 初期値を指定する。

<<Max Iter(int) 実行される反復の最大回数を指定する整数。

<<Tolerance(p) *p* は収束基準値。デフォルト値は 10^5。

<<Show Details("true") 結果のリスト（目標値、反復回数、勾配、ヘッセ行列）を戻す。最適化のステップごとの結果をログに出力する。

Desirability(yVector, desireVector, y)

説明

3 点を通過するという条件のもと、応答変数（*y*）の満足度を決定する関数を定義する。*yVector* と *desireVector* は、満足度関数を決定する 3 点に対応するベクトルです。実際の関数は、満足度値が大きい方が良いか、小さい方が良いか、目標値に合わせるか、または目標値がないかのいずれであるかによって異なります。

戻り値

満足度関数

引数

yVector 3 つの入力値

desireVector 対応する 3 つの満足度の値

y 満足度を計算する値

LPSolve(A, b, c, L, U, neq, nle, nge, <slackVars(Boolean)>)

説明

戻り値のリストの第 1 要素は、決定変数の値（slackVars に 1 が指定された場合にはスラック変数の値も含む）。第 2 要素は、目的変数の最適値（存在する場合のみ）。

引数

A 制約式の係数を表す行列

b 制約式の右辺値を列にした行列

c 目的関数のコスト係数のベクトル

L, U 下限値と上限値を表す行列
neq 等号制約式の数
nle「以下」を示す不等号制約式の数
nge「以上」を示す不等号制約式の数
slackVars(Boolean)（オプション）決定変数に加えて、スラック変数も戻すかどうかを指定する。デフォルト値は 0 です。

ノート

制約式は、等号制約、「以下」を示す不等号制約、「以上」を示す不等号制約の順に指定しなければなりません。

Maximize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

説明

式 *expr* を最大化する引数 *x* の値を求める。引数はリストで指定すること。引数名の後に括弧で囲んだ数値を指定することにより、各引数の下限と上限を指定できる。さらに、反復の最大回数、収束基準、最適化の履歴表示も指定できる。ヘッセ行列について解析的微分が行える場合は、Newton-Raphson 法が使用される。それ以外の場合は、対称ランクワン法（SR1）に基づく準 Newton 法が使用される。

メッセージ

<<Max Iter(int) 実行される反復の最大回数を指定する整数。
<<Tolerance(p) *p* は収束基準値。デフォルト値は 10^-8。
<<Show Details("true") 結果のリスト（目標値、反復回数、勾配、ヘッセ行列）を戻す。最適化のステップごとの結果をログに出力する。

Minimize(expr, {x1(low1, up1), x2(low2, up2), ...}, messages)

説明

式 *expr* を最小化する引数リスト *x* の値を求める。各引数の下限および上限を、括弧内に指定できる。また、反復の最大回数、収束基準値、最適化の詳細表示を指定する引数もある。ヘッセ行列について解析的微分が行える場合は、Newton-Raphson 法が使用される。それ以外の場合は、対称ランクワン法（SR1）に基づく準 Newton 法が使用される。

メッセージ

<<Max Iter(int) 実行される反復の最大回数を指定する整数。
<<Tolerance(p) *p* は収束基準値。デフォルト値は 10^-8。
<<Show Details("true") 結果のリスト（目標値、反復回数、勾配、ヘッセ行列）を戻す。最適化のステップごとの結果をログに出力する。

確率関数

Beta Density(x, alpha, beta, <theta>, <sigma>)

説明

ベータ分布の密度を戻す。

戻り値

指定されたパラメータをもつベータ分布の分位点 *x* における密度

引数

x (theta,theta+sigma) の範囲の分位点。theta のデフォルト値は 0 です。sigma のデフォルト値は 1 です。

alpha, beta 形状パラメータ。両者とも正の値。

theta（オプション）しきい値。範囲は $-\infty < \theta < \infty$。デフォルト値は 0

sigma（オプション）尺度パラメータ。正の値。デフォルト値は 1

ノート

ベータ分布は、分位点 x の定義域が無限であるような正規分布やガンマ分布とは異なり、限定された区間に対してだけ正の密度を持ちます。ベータ分布は、割合のような、範囲が 0 から 1 までに制限されている確率変数をモデル化する場合に役立ちます。

Beta Distribution(x, alpha, beta, <theta>, <sigma>)

説明

ベータ分布の累積分布関数の値を戻す。Beta Distribution() 関数は、Beta Quantile() 関数の逆関数です。指定された形状パラメータ（α と β）を持つベータ分布の累積分布関数で、ベータ分布に従う確率変数が分位点以下になる確率を戻します。

戻り値

指定されたパラメータをもつベータ分布の分位点 *x* における累積分布関数の値

引数

x (theta,theta+sigma) の範囲の分位点

alpha, beta 形状パラメータ。両者とも正の値。

theta（オプション）しきい値。範囲は $-\infty < \theta < \infty$。デフォルト値は 0

sigma（オプション）尺度パラメータ。正の値。デフォルト値は 1

Beta Quantile(p, alpha, beta, <theta>, <sigma>)

説明

ベータ分布の分位点を戻す。Beta Quantile() 関数は、Beta Distribution() 関数の逆関数です。

戻り値

指定されたパラメータをもつベータ分布の累積確率 *p* に対する分位点

引数

p 下側累積確率。*p* は 0 ～ 1 の間でなければならない

alpha, beta 形状パラメータ。両者とも正の値。

theta（オプション）しきい値。範囲は $-\infty < \theta < \infty$。デフォルト値は 0

sigma（オプション）尺度パラメータ。正の値。デフォルト値は 1

Cauchy Density(q, , <scale>)

説明

中心 *mu*、尺度 *sigma* の Cauchy 分布の *q* における密度を戻す。

Cauchy Distribution(q, , <scale>)

説明

Cauchy 分布に従う確率変数が *q* 以下になる確率を戻す。

Cauchy Quantile(p, , <scale>)

説明

Cauchy 分布の下側累積確率が *p* となる分位点を戻す。

ChiSquare Density(q, df,)

説明

自由度を *df*、オプションの非心度パラメータを *center* としたカイ 2 乗分布の、分位点 *q* における密度を戻す。

戻り値

カイ 2 乗分布の密度

引数

q 分位点
df 自由度
center 非心度パラメータ

ChiSquare Distribution(q, df,)

説明

自由度を df、オプションの非心度パラメータを center としたカイ 2 乗分布の、分位点 q における累積分布関数の値を戻す。

ChiSquare Log CDistribution(x, df, <nc>)

説明

カイ 2 乗分布の分位点 x での上側累積確率の対数を戻す。「1- 下側累積確率」の対数。

ChiSquare Log Density(x, df, <nc>)

説明

カイ 2 乗分布の分位点 x での密度関数の値の対数を戻す。

ChiSquare Log Distribution(x, df, <nc>)

説明

カイ 2 乗分布の分位点 x での累積分布関数の値の対数を戻す。

ChiSquare Noncentrality(x, df, prob)

説明

prob=ChiSquare Distribution (x, df, nc) が成立するような非心度 nc を求める。

ChiSquare Quantile(q, df,)

説明

自由度を df、オプションの非心度パラメータを center としたカイ 2 乗分布の、累積確率 *p* に対する分位点を戻す。

Dunnett P Value(q, nTrt, dfe, lambdaVec)

説明

Dunnett の多重比較検定の *p* 値を戻す。

戻り値

p 値を示す数値

引数

`q` 検定統計量の数値
`nTrt` 対照群と比較する処置群の数
`dfe` 誤差の自由度
`lambdaVec` 相関構造を表すパラメータのベクトル。lambdaVec が欠測値（.）の場合、ベクトルのすべての要素は、`1/Sqrt(2)` に設定されます。

Dunnett Quantile(1-alpha, nTrt, dfe, lambdaVec)

説明

Dunnett の多重比較検定に必要な分位点を戻す。

戻り値

分位点を示す数値

引数

`1-alpha` 信頼水準を示す数値
`nTrt` 対照群と比較する処置群の数
`dfe` 誤差の自由度
`lambdaVec` 相関構造を表すパラメータのベクトル。lambdaVec が欠測値（.）の場合、ベクトルのすべての要素は、`1/Sqrt(2)` に設定されます。

F Density(x, dfnum, dfden,)

説明

分子の自由度を *dfnum*、分母の自由度を *dfden*、オプションの非心度パラメータを *center* とした F 分布の分位点 *x* における密度を戻す。

F Distribution(x, dfnum, dfden,)

説明

分子の自由度を *dfnum*、分母の自由度を *dfden*、オプションの非心度パラメータを *center* とした F 分布の分位点 *x* における累積分布関数の値を戻す。

F Log CDistribution(x, dfn, dfd, <nc>)

説明

F 分布の分位点 x における上側累積確率の対数を戻す。

F Log Density(x, dfn, dfd, <nc>)

説明

F 分布の分位点 x での密度関数の値の対数を戻す。

F Log Distribution(x, dfn, dfd, <nc>)

説明

F 分布の分位点 x における累積分布関数の値の対数を戻す。

F Noncentrality(x, ndf, ddf, prob)

説明

prob=F Distribution (x, ndf, ddf, nc) が成立するような非心度 nc を求める。

F Power(alpha, dfh, dfm, d, n)

説明

与えられた引数から、*F* 検定や *t* 検定の検出力を計算する。

F Quantile(x, dfnum, dfden,)

説明

分子の自由度を *dfnum*、分母の自由度を *dfden*、オプションの非心度パラメータを *center* とした F 分布の累積確率 *p* に対する分位点を戻す。

F Sample Size(alpha, dfh, dfm, d, power)

説明

与えられた引数から、*F* 検定や *t* 検定の標本サイズを計算する。

Frechet Density(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の Fréchet 分布の *x* における密度を戻す。

引数

x 数値
mu 位置
sigma 尺度

Frechet Distribution(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の Fréchet 分布の *x* における下側累積確率を戻す。

引数

x 数値
mu 位置
sigma 尺度

Frechet Quantile(p, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の Fréchet 分布の累積確率 *p* における分位点を戻す。

引数

p 下側累積確率。*p* は 0 ～ 1 の間でなければならない
mu 位置
sigma 尺度

Gamma Density(q, alpha, <scale>, <threshold>)

説明

ガンマ分布の q における密度を戻す。

戻り値

指定されたパラメータをもつガンマ分布の分位点 *q* における密度関数の値

引数

q 分位点
alpha 形状パラメータ。正の値
scale（オプション）尺度パラメータ。正の値。デフォルト値は 1
threshold（オプション）閾値パラメータ。範囲は $-\infty < \theta < \infty$。デフォルト値は 0

Gamma Distribution(x, <shape, scale, threshold>)

IGamma(x, <shape, scale, threshold>)

説明

形状パラメータ（*shape*）、尺度パラメータ（*scale*）、閾値（*threshold*）が与えられた値をとるときのガンマ分布で、分位点 *x* における累積分布関数の値を戻す。

Gamma Log CDistribution(x, alpha, <scale = 1>, <threshold = 0>)

説明

計算できる範囲がはるかに大きいことを除けば、Log (1 - Gamma Distribution(x, alpha)) と同じ。

Gamma Log Density(x, alpha, <scale = 1>, <threshold = 0>)

説明

計算できる範囲がはるかに大きいことを除けば、Log(GammaDensity(x, alpha)) と同じ。

Gamma Log Distribution(x, alpha, <scale = 1>, <threshold = 0>)

説明

計算できる範囲がはるかに大きいことを除けば、Log(Gamma Distribution(x, alpha)) と同じ。

Gamma Quantile(p, <shape, scale, threshold>)

説明

形状パラメータ（*shape*）、尺度パラメータ（*scale*）、閾値（*threshold*）が与えられた値をとるときのガンマ分布で、累積確率 *p* に対する分位点を戻す。

GenGamma Density(x, mu, sigma, lambda)

説明

拡張一般化ガンマ分布の密度関数。パラメータが *mu*、*sigma*、*lambda* の拡張一般化ガンマ確率分布の *x* での密度を戻す。

GenGamma Distribution(x, mu, sigma, lambda)

説明

拡張一般化ガンマ分布の累積分布関数。パラメータが *mu*、*sigma*、*lambda* の拡張一般化ガンマ分布に基づく確率変数が *x* より小さい確率を戻す。

GenGamma Quantile(p, mu, sigma, lambda)

説明

拡張一般化ガンマ分布の分位点関数。パラメータが mu、sigma、lambda の拡張一般化ガンマ分布の下側累積確率が *p* となる分位点を戻す。

GLog Density(q, mu, sigma, lambda)

説明

一般化対数分布の密度関数。位置 *mu*、尺度 *sigma*、形状 *lambda* の一般化対数分布の *q* における密度を戻す。

GLog Distribution(q, mu, sigma, lambda)

説明

一般化対数分布の累積分布関数。一般化対数分布に従う確率変数が *q* 以下になる確率を戻す。

GLog Quantile(p, mu, sigma, lambda)

説明

分布の下側累積確率が *p* となる分位点を戻す。

IGamma()

「Gamma Distribution(x, <shape, scale, threshold>)」（165ページ）を参照してください。

Johnson Sb Density(q, gamma, delta, theta, sigma)

説明

Johnson Sb 分布の *q* における密度を戻す。

引数

q (*theta*,*theta*+*sigma*) の範囲の値
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 尺度パラメータ。正の値

Johnson Sb Distribution(q, gamma, delta, theta, sigma)

説明

Johnson Sb 分布に従う確率変数が *q* 以下になる確率を戻す。

引数

q (*theta*,*theta*+*sigma*) の範囲の値
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 尺度パラメータ。正の値

Johnson Sb Quantile(p, gamma, delta, theta, sigma)

説明

分布の下側累積確率が *p* となる分位点を戻す。

引数

p 下側累積確率。*p* は 0 ～ 1 の間でなければならない
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 尺度パラメータ。正の値

Johnson Sl Density(q, gamma, delta, theta, sigma)

説明

Johnson Sl 分布の *q* における密度を戻す。

引数

q sigma が +1 のときは (theta,+ ∞)、sigma が -1 のときは (- ∞ ,theta) の範囲の値
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 分布が正の方向に歪むか、負の方向に歪むかを定義するパラメータ。Sigma は、+1（正の方向）または -1（負の方向）のどちらかをとります。

Johnson Sl Distribution(q, gamma, delta, theta, sigma)

説明

Johnson Sl 分布に従う確率変数が *q* 以下になる確率を戻す。

引数

q sigma が +1 のときは (theta,+ ∞)、sigma が -1 のときは (- ∞ ,theta) の範囲の値
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 分布が正の方向に歪むか、負の方向に歪むかを定義するパラメータ。Sigma は、+1（正の方向）または -1（負の方向）のどちらかをとります。

Johnson Sl Quantile(p, gamma, delta, theta, sigma)

説明

分布の下側累積確率が *p* となる分位点を戻す。

引数

p 下側累積確率。*p* は 0 ～ 1 の間でなければならない
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値

theta 位置パラメータ。任意の値
sigma 分布が正の方向に歪むか、負の方向に歪むかを定義するパラメータ。Sigmaは、+1（正の方向）または-1（負の方向）のどちらかをとります。

Johnson Su Density(q, gamma, delta, theta, sigma)

説明

Johnson Su 分布の *q* における密度を戻す。

引数

q (-∞,+∞) の範囲の値
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 尺度パラメータ。正の値

Johnson Su Distribution(q, gamma, delta, theta, sigma)

説明

Johnson Su 分布に従う確率変数が *q* 以下になる確率を戻す。

引数

q (-∞,+∞) の範囲の値
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 尺度パラメータ。正の値

Johnson Su Quantile(p, gamma, delta, theta, sigma)

説明

分布の下側累積確率が *p* となる分位点を戻す。

引数

p 下側累積確率。*p* は 0 ～ 1 の間でなければならない
gamma 形状パラメータ。任意の値
delta 形状パラメータ。正の値
theta 位置パラメータ。任意の値
sigma 尺度パラメータ。正の値

LEV Density(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の最大極値分布の *x* における密度を戻す。

LEV Distribution(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の最大極値分布の *x* における下側累積確率を戻す。

LEV Quantile(p, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の最大極値分布の累積確率 *p* における分位点を戻す。

LogGenGamma Density(x, mu, sigma, lambda)

説明

対数一般化ガンマ分布の密度関数。パラメータが *mu*、*sigma*、*lambda* の対数一般化ガンマ確率分布の *x* での密度を戻す。

LogGenGamma Distribution(x, mu, sigma, lambda)

説明

対数一般化ガンマ分布の累積分布関数。パラメータが *mu*、*sigma*、*lambda* の対数一般化ガンマ分布に基づく確率変数が *x* より小さい確率を戻す。

LogGenGamma Quantile(p, mu, sigma, lambda)

説明

対数一般化ガンマ分布の分位点関数。パラメータが *mu*、*sigma*、*lambda* の対数一般化ガンマ分布の下側累積確率が *p* となる分位点を戻す。

Logistic Density(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* のロジスティック分布の *x* における密度を戻す。

Logistic Distribution(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* のロジスティック分布の *x* における下側累積確率を戻す。

Logistic Quantile(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* のロジスティック分布の累積確率 *p* における分位点を戻す。

Loglogistic Density(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の対数ロジスティック分布の *x* における密度を戻す。

Loglogistic Distribution(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の対数ロジスティック分布の *x* における下側累積確率を戻す。

Loglogistic Quantile(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の対数ロジスティック分布の累積確率 *p* における分位点を戻す。

Lognormal Density(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の対数正規分布の *x* における密度を戻す。

Lognormal Distribution(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の対数正規分布の *x* における下側累積確率を戻す。

Lognormal Quantile(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の対数正規分布の *p* における分位点を戻す。

Normal Biv Distribution(x, y, r, <mu1>, <s1>, <mu2>, <s2>)

説明

2 変量正規分布に従う確率変数 (X, Y) が (*x, y*) 以下になる確率を計算する。ここで、X の周辺分布は平均 *mu1*、標準偏差 *s1*、Y の周辺分布は平均 *mu2*、標準偏差 *s2* の正規分布に従っているとします。また、2 変量の相関関数は *r* とします。*mu1*、*s1*、*mu2*、*s2* が指定されていない場合、*mu1*=0、*s1*=1、*mu2*=0、*s2*=1 の標準正規分布が使われます。

Normal Density(x, <mean>, <stddev>)

説明

平均（*mean*）、標準偏差（*stddev*）を持つ正規分布の、分位点 *x* における密度関数の値を戻す。デフォルトの平均（*mean*）は 0、デフォルトの標準偏差（*stddev*）は 1 です。

Normal Distribution(x, <mean>, <stddev>)

説明

平均（*mean*）、標準偏差（*stddev*）を持つ正規分布の、分位点 *x* における累積分布関数の値を戻す。デフォルトの平均（*mean*）は 0 です。デフォルトの標準偏差（*stddev*）は 1 です。

Normal Log CDistribution(x)

説明

正規分布の分位点 *x* での上側累積確率の対数を戻す。

Normal Log Density(x)

説明

正規分布の分位点 *x* での密度関数の値の対数を戻す。

Normal Log Distribution(x)

説明

正規分布の分位点 *x* での累積分布関数の値の対数を戻す。

Normal Mixture Density(q, mean, stdev, probability)

説明

正規混合分布の密度関数。グループ平均 *mean*、グループ標準偏差 *stdev*、グループ確率 *probability* の *q* における密度を戻す。*mean*、*stdev*、および *probability* はすべて同じサイズのベクトルです。

Normal Mixture Distribution(q, mean, stdev, probability)

説明

正規混合分布の確率関数。グループ平均 *mean*、グループ標準偏差 *stdev*、グループ確率 *probability* の正規混合分布に従う確率変数が *q* よりも小さい確率を戻す。*mean*、*stdev*、および *probability* はすべて同じサイズのベクトルです。

Normal Mixture Quantile(p, mean, stdev, probability)

説明

正規混合分布の分位点関数。正規混合分布の下側累積確率が p となる分位点を戻す。*mean*、*stdev*、および *probability* はすべて同じサイズのベクトルです。

Normal Quantile(p, <mean>, <stddev>)

Probit(p, <mean>, <stddev>)

説明

平均（*mean*）、標準偏差（*stddev*）をもつ正規分布の、累積確率 *p* に対する分位点を戻す。デフォルトの平均（*mean*）は 0 です。デフォルトの標準偏差（*stddev*）は 1 です。

Probit()

「Normal Quantile(p, <mean>, <stddev>)」（172ページ）を参照してください。

SEV Density(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の最小極値分布の *x* における密度を戻す。

SEV Distribution(x, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の最小極値分布の *x* における下側累積確率を戻す。

SEV Quantile(p, mu, sigma)

説明

位置 *mu*、尺度 *sigma* の最小極値分布の *p* における分位点を戻す。

Students t Density()

「t Density(q, df)」（172ページ）を参照してください。

Students t Distribution()

「t Distribution(q, df, <nonCentrality>)」（173ページ）を参照してください。

Students t Quantile()

「t Quantile(p, df, <nonCentrality>)」（173ページ）を参照してください。

t Density(q, df)

Students t Density(q, df)

説明

Student の t 分布の密度関数。指定された自由度（*df*）をもつ Student の t 分布の、分位点 *x* における密度関数の値を戻す。

t Distribution(q, df, <nonCentrality>)

Students t Distribution(q, df, <nonCentrality>)

説明

Student の t 分布の累積分布関数。Student の *t* 分布に従う確率変数が *q* 以下になる確率を戻す。*NonCentrality* のデフォルトは 0 です。

t Log CDistribution(x, df, <nc>)

説明

t 分布の分位点 *x* での上側累積確率の対数を戻す。

t Log Density(x, df, <nc>)

説明

t 分布の分位点 *x* での密度関数の値の対数を戻す。

t Log Distribution(x, df, <nc>)

説明

t 分布の分位点 *x* での累積分布関数の値の対数を戻す。

t Noncentrality(x, df, prob)

説明

次の式が成り立つような非心度を求める。

prob=T Distribution (*x*, *df*, *nc*)

t Quantile(p, df, <nonCentrality>)

Students t Quantile(p, df, <nonCentrality>)

説明

Student の t 分布の分位点関数。指定された自由度（*df*）をもつ Student の *t* 分布の、下側累積確率が *p* となる分位点を戻す。*NonCentrality* のデフォルトは 0 です。

Tukey HSD P Value(q, n, dfe)

説明

Tukey の HSD 多重比較検定の *p* 値を戻す。

引数

q 検定統計量

n 検定されるグループの数

dfe すべての標本から計算される誤差の自由度

Tukey HSD Quantile(1-alpha, n, dfe)

説明

Tukey の HSD 多重比較検定に必要な分位点を戻す。

引数

1-alpha 信頼水準
n 検定されるグループの数
dfe すべての標本から計算される誤差の自由度

Weibull Density(x, shape, <scale, threshold>)

説明

指定されたパラメータをもつ Weibull 分布の、分位点 *x* における密度を戻す。

Weibull Distribution(x, shape, <scale, threshold>)

説明

指定されたパラメータをもつ Weibull 分布の、分位点 *x* における累積分布関数の値を戻す。

Weibull Quantile(p, shape, <scale, threshold>)

説明

指定されたパラメータをもつ Weibull 分布の、累積確率 *p* に対する分位点を戻す。

プログラミング関数

As Column(name)

As Column(dt, name)

:name

dt:name

説明

このスコープ演算子により、グローバル変数ではなく、現在のデータテーブル（または、オプションのデータテーブル参照引数 *dt* で指定したテーブル）内のデータテーブル列として変数（*name*）が評価される。

引数

name 変数名
dt データテーブル参照

ノート

:name は、現在のデータテーブルの列名を参照します。dt:name を使用して参照先データテーブルを指定することもできます。

As Constant(expr)

説明
計算後に変化しない値を生成するために、式を一度評価する。

戻り値
評価の結果

引数
expr 任意の JSL 式

ノート
データテーブルに予測式の列を保存できるいくつかのプラットフォームでは、As Constant() が使用されています。この関数は、計算式の一部がすべての行について一定である場合に挿入されます。1 行目について引数を評価し、後続の行については、再評価せずに同じ結果を使用します。

As Global(name)

::name

説明
このスコープ演算子により、データテーブル列ではなく、グローバル変数として変数（*name*）が評価される。

引数
name 変数名

As List(matrix)

「As List(matrix)」（126ページ）を参照してください。

As Name(string)

説明
引数を文字列として評価し、それを名前に変更する。

戻り値
名前

As Namespace(name)

説明
指定された名前空間にアクセスする。そのような名前空間が存在しない場合はエラーが戻されます。

戻り値
名前空間

引数
name 定義された引用符なしの名前空間

As Scoped(namespace, variable)

namespace:variable

説明

指定の名前空間 *namespace* にある指定の変数 *variable* にアクセスする。

戻り値

変数の値、ただしスコープ変数が見つからない場合はエラー

引数

namespace 定義された名前空間の名前
variable 名前空間 *namespace* 内の定義された変数

Associative Array({key, value}, ...)

Associative Array(keys, values)

説明

連想配列（「辞書」または「ハッシュマップ」ともいう）を作成する。

戻り値

連想配列オブジェクト

引数

キー（key）と値（value）をペアとしたリスト。または、キーと値それぞれに、リスト、行列、またはデータテーブル列を指定することもできます。

Clear Globals(<name>, <name>, ...)

説明

すべてのグローバルシンボルの値をクリアする。グローバル以外の任意のスコープにおけるシンボルには影響しません。1つまたは複数の名前が指定されている場合は、それらのグローバルシンボルだけがクリアされます。

戻り値

なし

引数

name（オプション）任意のグローバル変数名

関連項目

「Clear Symbols(<name>, <name>, ...)」(177 ページ)

Clear Log()

説明

ログウィンドウの内容をクリアする。

Clear Symbols(<name>, <name>, ...)

説明

任意およびすべてのスコープにおけるすべてのシンボルの値をクリアする。1つまたは複数の名前が指定されている場合は、それらのシンボルだけがクリアされます。

戻り値

なし

引数

name（オプション）任意のグローバル変数名

関連項目

「Clear Globals(<name>, <name>, ...)」（176ページ）。

Close Log()

説明

ログを閉じる。

Delete Globals(<name>, <name>, ...)

説明

グローバルシンボルをすべて削除する（ロックされているもの以外）。グローバル以外の任意のスコープにおけるシンボルには影響しません。1つまたは複数の名前が指定されている場合は、それらのグローバルシンボルだけがクリアされます。

引数

name（オプション）任意のグローバル変数名

関連項目

「Delete Symbols(<name>, <name>, ...)」（177ページ）を参照してください。

Delete Symbols(<name>, <name>, ...)

説明

任意およびすべてのスコープにおけるすべてのシンボルを削除する。1つまたは複数の名前が指定されている場合は、それらのシンボルだけが削除されます。

引数

name（オプション）任意のグローバル変数名

関連項目

「Delete Globals(<name>, <name>, ...)」（177ページ）。

Eval(expr)

説明

expr を評価し、その評価の結果を戻す（つまり、アンクォートを行う）。

戻り値
評価の結果

引数
expr 任意の JSL 式

Eval Insert(string, <startDel>, <endDel>, < <<Use Locale(1) >)

説明
複数の置換を行う。

戻り値
結果

引数
string 置換したい式が途中に指定されている引用符付き文字列
startDel（オプション）開始の区切り文字 デフォルトの値は ^
endDel（オプション）終了の区切り文字。デフォルトの終了文字は、開始文字と同じ文字
Use Locale(1)（オプション）ロケール固有の数値形式を維持する引数

Eval Insert Into(string, <startDel>, <endDel>)

説明
複数の置換を行う。EvalInsert と同じ処理が実行され、*string* に結果が割り当てられます。

戻り値
結果

引数
string 式が組み込まれている文字列を含む文字列変数
startDel（オプション）開始の区切り文字 デフォルトの値は ^
endDel（オプション）終了の区切り文字。デフォルトの終了文字は、開始文字と同じ文字

Eval List

「Eval List(list)」（127ページ）を参照してください。

Exit(<NoSave>)

Quit(<NoSave>)

説明
JMP を終了する。

戻り値
なし

引数

NoSave（オプション）名前付きコマンド。開いているファイルを保存するかどうかを確認せずに、JMPを終了します。このオプション名も、大文字／小文字の区別は**なく**、また、スペースを含めてもかまいません。

First(expr, <expr>, ...)

説明

引数で指定されたすべての式を評価する。

戻り値

最初に評価された式の結果のみ

引数

expr 任意の有効なJSL式

Function({arguments}, <{local variables}>, <Return(<expr>)>, script)

説明

*arguments*をローカル変数とした、スクリプト (*script*) を保存する。

戻り値

定義されたとおりの関数。Return() 引数が指定されている場合は、指定の式を戻します。

呼ばれたときは、指定の引数（*arguments*）でスクリプト（*script*）を実行した結果を戻します。

引数

{arguments} 関数に渡す引数のリスト。オプションまたは必須の引数を指定できます。

{local variables} 関数に対してローカルである変数のリスト。ローカル変数は、次の3つの方法で宣言できます。

```
{var1, var2}
{var1=0, var1="a string"}
{Default Local}
```

最後のオプションは、関数内で使用されている変数のうち、適用範囲が指定されていないものがすべてローカルであることを宣言します。

Return(expr)（オプション）ユーザー定義関数から式を戻す引数。ヌルの式を使用した場合は、ピリオド（.）が戻されます。

script 任意の有効なJSLスクリプト

Get Environment Variable("variable")

説明

オペレーティングシステムの環境変数の値を取得する。

戻り値

指定された環境変数の値を示す文字列。指定された変数が見つからない場合は空を戻します。

引数

"variable" 環境変数の名前を示す文字列

ノート

Macintosh では環境変数名に大文字／小文字の区別があります。Windows では区別がありません。

Get Log(<n>)

説明

ログの内容を、リストで戻す。

戻り値

文字列のリスト。各文字列には 1 行分のログが含まれます。

引数

n （オプション）整数。引数が指定されなかった場合は、すべての行を戻します。正の数が指定された場合は、最初の *n* 行を戻します。負の数が指定された場合、最後の *n* 行を戻します。*n*=0 の場合、どの行も戻しません（空のリストを戻します）。ログが空の場合、空のリストを戻します。

Include("pathname", <named arguments>)

説明

引用符付き文字列として指定されたパス名（*pathname*）に対応するスクリプトファイルを開き、スクリプトを実行する。

戻り値

インクルードされたスクリプトが戻すものすべて。<<Parse Only オプションを使用した場合、Include はスクリプトの中身を戻します。

名前付き引数

<<Parse Only スクリプトを解析するが、実行はしない。

<<New Context インクルードしたスクリプトを、独自の名前空間で実行する。親とインクルードされるスクリプトがグローバル名前空間を使用する場合は、<<Names Default to Here を <<New Context とともに指定します。

<<Allow Include File Recursion インクルード元のファイルを、自分自身からインクルードすることを許す

例

```
::x = 5;
::Dice = 1000;
Show Symbols(); // x = 5;
Include( "$SAMPLE_SCRIPTS/scriptInclude.jsl", <<New Context, <<Names Default To
   Here);
Show Symbols();
```

scriptInclude.jsl によって x が 1 に設定されるので、x の結果は 1 になります。

Include File List()

説明

実行時にインクルードされているファイルのリストを戻す。

Is Log Open()

説明

ログウィンドウが開いているかどうかの結果を戻す。

list

「List(a, b, c, ...)」（128ページ）を参照してください。

Local({name=value, ...}, script)

説明

変数（*name*）をローカル変数として定義し、スクリプト（script）を実行する。

Local Here(expression)

説明

ローカルの Here 名前空間ブロックを作成する。複数のスクリプトが同じルートの名前空間から実行されるときの競合を回避したい場合に、この関数を使用します（たとえば、スクリプトが同じ変数を持つ 2 つのボタンスクリプトを実行する場合など）。引数には、任意の有効な JSL 式を指定できます。

Lock Globals(name1, name2, ...)

説明

1 つ以上のグローバル変数を、変更されないようにロックする。

Lock Symbols(<name>, <name>, ...)

説明

指定のグローバル変数をロックする。これにより、変数が変更されたり消去されたりするのを防げます。変数が指定されていない場合は、すべてのグローバル変数をロックします。なお、変数が指定されていない場合でも、スクリプトで **Names Default To Here** モードがオンになっている場合は、すべてのローカル変数だけをロックします。

LogCapture(expr)

説明

引数 expr を評価し、通常ログに送られるはずの出力を取得し、ログに出力する代わりに文字列で戻す。

戻り値

ログ出力を示す文字列

引数

任意の有効な JSL 式

ノート

ログには何も出力されません。

N Items

「N Items(source)」（128ページ）を参照してください。

Names Default To Here(Boolean)

説明

未解決の名前を保持する場所を指定する。グローバルまたはローカルに保持する場合は *Boolean* を 0、Here スコープ内に保持する場合は *Boolean* を 1 に指定します。

Namespace(name)

説明

指定された名前（*name*）の名前空間への参照を戻す。

引数

name 名前空間名の文字列、または、名前空間への参照

Namespace Exists(name)

説明

指定された名前（*name*）の名前空間が存在する場合は 1、そうでない場合は 0 を戻す。

New Namespace(<"name">, <{expr, ...}>)

説明

指定の名前（name）で新しい名前空間を作成する。名前が指定されていない場合は、匿名の名前が使用されます。

戻り値

名前空間への参照

引数

name（オプション）新しい名前空間の名前を示す引用符付き文字列
{list of expressions}（オプション）名前空間内の式のリスト

Open Log()

説明

ログを開く。ログウィンドウがすでに開いている場合、ブール値の引数（1）を指定するとウィンドウがアクティブになります。

Parameter({name=value, ...}, model expression)

説明

「非線形回帰」プラットフォームで用いるモデルの計算式パラメータを定義する。

Parse(string)

説明

文字列をJSLの式に変換する。

Print(expr, expr, ...)

説明

指定された式（*expr*）の値をログに出力する。

Quit()

「Exit(<NoSave>)」（178ページ）を参照してください。

Recurse(arg1,arg2,...)

説明

定義した*関数*の再帰呼び出しを行う。

Save Log(pathname)

説明

指定のファイルにログの内容を書き込む。

Send(obj, message)

obj << message

説明

プラットフォームオブジェクト（*obj*）にメッセージ（*message*）を送る。

Set Environment Variable("variable", <"value">)

説明

環境変数およびその値に設定する。"value" 引数が未指定または空文字列の場合、環境変数はJMPプロセスの環境変数テーブルから削除されます。

Show(expr, expr, ...)

説明

各式（*expr*）の名前と値をログに出力する。

Show Globals()

説明

すべてのグローバルシンボルの値を表示する。グローバル以外の任意のスコープにおけるシンボルは表示されません。

関連項目

「Show Symbols()」(184 ページ)。

Show Namespaces()

説明

ユーザが定義した名前空間（名前付きおよび匿名の両方）の内容を表示する。

Show Symbols()

説明

すべてのスコープにおけるすべてのシンボルの値を表示する。

関連項目

「Show Globals()」(183 ページ)。

Sort List

「Sort List(list|expr)」(130ページ）を参照してください。

Sort List Into

「Sort List Into(list|expr)」(130ページ）を参照してください。

Throw("text")

説明

例外をスローする。テキスト（*text*）を指定した場合、Throw が実行されると、exception_msg という名前のグローバル変数にその *text* が格納されます。感嘆符（!）で始まる *text* の場合は、どこで例外が発生したかを示す診断エラーメッセージを生成します。

Try(expr1, expr2)

説明

最初の式（*expr1*）を評価して、例外がスローされた場合には、そこで停止し、2 番目の式（*expr2*）を評価してその結果を戻す。例外がスローされなかった場合は、expr2 は評価されない。

例

```
Try( Sqrt( "s" ), "invalid" );
  "invalid"

Try( Sqrt( "s" ), exception_msg );
  {"引数を数値(または行列)に変換できません。"(1, 2, "Sqrt", Sqrt/*###*/("s"))}
```

ノート

Expr2 には、文字列を指定することも、戻されたエラーに関する情報を含むグローバルな例外メッセージ（exception_msg）を指定することもできます。

Type(x)

説明

引数（*x*）のタイプを示す文字列を戻す 戻されるタイプとしては、Unknown、List、DisplayBox、Picture、Column、TableVar、Table、Empty、Pattern、Date、Integer、Number、String、Name、Matrix、RowState、Expression、Associative Array、Blob があります。

Unlock Symbols(name1, name2, ...)

Unlock Globals(name1, name2, ...)

説明

Lock Symbols() または Lock Globals() コマンドでロックされた指定の変数のロックを解除する。

Wait(n)

説明

n 秒待ってから、スクリプトの実行を継続する。

Watch(all | name1, ...)

説明

変数（グローバル、ローカル、名前空間内の変数）およびその値をウィンドウに表示する。引数に "all" が指定されている場合、すべての変数がウィンドウに表示されます。

ノート

新しく追加した変数は、ウィンドウに追加されません。

メッセージを使って変更された連想配列の表示はサポートされていません。

Wild()

説明

どんな式にも一致するワイルドカードの位置を表す。この関数は、Extract Expr() での式のマッチングにだけ使用できる。

Wild List()

説明

どんな式にも一致するワイルドカードの位置を表す。この関数は、Extract Expr() での式のマッチングにだけ使用できる。

Write("text")

説明

テキスト（*text*）から引用符を取り除いたものをログに書き込む。

Rインテグレーション関数

R Connect(<named_arguments>)

説明

現在のR接続オブジェクトを戻す。Rへの接続が確立されていない場合は、R接続インターフェースを初期化し、アクティブなRインターフェース接続をスクリプト可能なオブジェクトとして戻します。

戻り値

スクリプト可能なオブジェクト

引数

Echo(Boolean)（オプション）実行したRのプログラムコードを、JMPのログに出力する。このオプションは、グローバルです。デフォルト値は1（真）。

R Execute({ list of inputs }, { list of outputs }, "rCode", <named_arguments>)

説明

現在のグルーバルなR接続に対して、第1引数のリストに指定されたJMP変数を送り、第3引数に指定されたRコードをサブミットする。第2引数のリストに指定されたR変数が、JMPに戻されます。

戻り値

成功した場合は0、そうでない場合は0

引数

{ list of inputs } 入力としてRに送られるJMP変数名のリスト

{ list of outputs } Rから戻される出力を格納するJMP変数名のリスト

rCode サブミットするRコードを示す引用符付き文字列

Expand(Boolean)（オプション）この引数が1（真）の場合、Rにサブミットする前に、Rコードに対してEval Insert()を実行します。

Echo(Boolean)（オプション）この引数が1（真）の場合、実行したRのプログラムコードを、JMPのログに出力する。このオプションは、グローバルです。デフォルト値は1（真）。

例

JMP変数xとyをRに送り、Rステートメントz <- x * yを実行し、そのR変数zをJMP側で取得します。

```
x = [1 2 3];
y = [4 5 6];
rc = R Execute( {x, y}, {z}, "z <- x * y" );
```

R Get(variable_name)

説明

Rの変数を、JMPで取得する。

戻り値
variable_name で指定した変数の値。

引数
name 必須。値を取得する R 変数の名前

例
qbx という名前の行列と、df という名前のデータフレームが、R 上に存在するとします。

```
// R 変数 qbx の値を取得し、それを JMP 変数 qbx に代入
qbx = R Get( qbx );

// R 変数 df のデータフレームを、JMP 側でデータテーブルとして取得。それを、JMP 変数 df で参照
df = R Get( df );
```

R Get Graphics("format")

説明
R のグラフウィンドウに最後に出力されたグラフを、指定の形式で取得する。

戻り値
JMP ピクチャーオブジェクト

引数
format 必須。使用するグラフィック形式。有効な形式は、"png"、"bmp"、"jpeg"、"jpg"、"tiff"、および "tif" です。

R Init(named_arguments)

説明
JMP から R への接続を、初期化する。

戻り値
成功した場合は 0、そうでない場合は 0 以外

引数
Echo(Boolean)（オプション）実行した R のプログラムコードを、JMP のログに出力する。このオプションは、グローバルです。デフォルト値は 1（真）。

R Is Connected()

説明
R への接続が存在するかどうかを特定する。

戻り値
アクティブな R 接続がある場合は 1、そうでない場合は 0 を戻す。

引数
なし

R JMP Name to R Name(name)

説明

R の命名規則に従い、JMP 変数名を、対応する R 変数名に変換する。

引数

name R に送る JMP 変数の名前

戻り値

R での命名規則に従った変数名を示す文字列

R Send(name, <R Name(name)>)

説明

JMP から R に変数を送る。

戻り値

成功した場合は 0、そうでない場合は 0

引数

name（必須）R に送る JMP 変数の名前

R Name(name)（オプション）R に送る変数に別の名前をつけることができる。次に例を示します。

```
R Send(Here:x, R Name("localx"))
```

以下は、データテーブルの場合のみ適用されます。

Selected(Boolean)（オプション）名前付き、ブール値。参照先データテーブルの選択された行のみを R に送ります。

Excluded(Boolean)（オプション）名前付き、ブール値。参照先データテーブルの除外された行のみを R に送ります。

Labeled(Boolean)（オプション）名前付き、ブール値。参照先データテーブルのラベルのついた行のみを R に送ります。

Hidden(Boolean)（オプション）名前付き、ブール値。参照先データテーブルの非表示の行のみを R に送ります。

Colored(Boolean)（オプション）名前付き、ブール値。参照先データテーブルの色のついた行のみを R に送ります。

Markered(Boolean)（オプション）名前付き、ブール値。参照先データテーブルのマーカーのついた行のみを R に送ります。

Row States(Boolean, <named arguments>)（オプション）名前付き。ブール値の引数とオプションの名前付き引数を指定する。JMP データテーブルにおける行属性の情報を、「RowState」という列名のデータ列を追加して、R に送ります。個別の設定をまとめて追加することによって、複数の行の属性を作成できます。個々の行属性を表す値は次のとおりです。

- Selected = 1
- Excluded = 2
- Hidden = 4
- Labeled = 8

- Colored = 16
- Markered = 32

Row States() 引数の名前付き引数は次のとおりです。

Colors(Boolean)（オプション）名前付き、ブール値。行の色を送ります。「RowStateColor」という名前のデータ列を追加します。

Markers(Boolean)（オプション）名前付き、ブール値。行のマーカーを送ります。「RowStateMarker」という名前のデータ列を追加します。

例

行列を変数 X に代入します。そして、その変数を R に送ります。

```
X = [1 2 3];
rc = R Send( X );
```

データテーブルを開き、その参照を *dt* に割り当てます。そして、そのデータテーブルを、現在の行属性も含めて R に送ります。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
rc = R Send( dt, Row States(1) );
```

R Send File("pathname", <R Name("name")>)

説明

JMP から R に指定したデータファイルを送る。

戻り値

成功した場合は 0、そうでない場合は 0

引数

pathname（必須）ファイルのパス名を含む引用符付き文字列。

R Name(name)（オプション）R に送るデータファイルに別の名前をつけることができる。

R Submit("rCode", <named_arguments>)

説明

指定された R コードを、R 上で実行する（現在のグローバルな R 接続に、サブミットする）。

戻り値

成功した場合は 0、そうでない場合は 0

引数

rCode（必須）サブミットする R コードを示す引用符付き文字列

Expand(Boolean)（オプション）この引数が 1（真）の場合、R にサブミットする前に、R コードに対して Eval Insert() を実行します。

Echo(Boolean)（オプション）この引数が 1（真）の場合、実行した R のプログラムコードを、JMP のログに出力する。このオプションは、グローバルです。デフォルト値は 1（真）。

Async(Boolean)（オプション）この引数が 1（真）の場合、Esc キーを押すか、または R 接続に対して rconn<<Control(Interrupt(1)) のメッセージを送ると、処理の途中でもキャンセルできる。デフォルト値は 0（偽）。

例

```
rc = R Submit("\[
   x <- rnorm(5)
   print(x)
   y <- rnorm(5)
   print(y)
   z = plot(x, y)
]\" );
```

R Submit File("pathname")

説明

pathname で指定されたファイルに含まれる R コードを、R 上で実行する。

戻り値

成功した場合は 0、そうでない場合は 0

引数

pathname 実行する R コードを含むファイルのパス名を示す引用符付き文字列

R Term()

説明

R への接続を終了する（アクティブな R 接続インターフェイスを終了する）。

戻り値

アクティブな R 接続がある場合は 1、そうでない場合は 0

引数

なし

乱数関数

Col Shuffle()

説明

評価のたびに値を無作為に並べ替える。

戻り値

最後の行に挿入された最後の値

引数

なし

例

```
For Each Row(x=:shuffle = colshuffle());
show(x)
  x = 6
```

上記のスクリプトは、データテーブルに 40 行ある場合、Shuffle という名前の列の各行に、1 ～ 40 の整数をランダムに挿入します。どの整数も、1 度ずつ使用されます。スクリプトが実行されるたびに、整数がランダムな順序で挿入されます。この例において、列 shuffle の 40 行目に挿入された整数が 6 の場合、その整数 6 が戻り値となります。

Random Beta(alpha, beta, <theta>, <sigma>)

説明

2 つの形状パラメータ *alpha* と *beta* のベータ分布に従う乱数を生成します。オプションで、下限値 *theta* と尺度パラメータ *sigma* を指定することができます。デフォルト値は *theta* = 0 および *sigma* = 1 です。

Random Beta Binomial(n, p, <delta>)

説明

試行回数 *n*、確率 *p*、相関 *delta* のベータ二項分布に従う乱数を戻す。*delta* のデフォルト値は 0 です。

Random Binomial(n, p)

説明

試行回数が *n* で、イベントの生起確率が *p* である二項分布に従う乱数を生成する。

Random Category(probA, resultA, ProbB, resultB, resultElse)

説明

交互に指定された確率と結果式に基づいて発生させたランダムなカテゴリを戻す。

Random Cauchy(<alpha>, <beta>)

説明

指定されたパラメータ *alpha*（位置）および *beta*（尺度）の Cauchy 分布に従う乱数を生成する。

Random ChiSquare(df, <noncentral>)

説明

指定された *df*（自由度）のカイ 2 乗分布に従う乱数を戻す。非心度パラメータの値はゼロ（デフォルト値）以上でなければなりません。

Random Exp()

説明

指数分布に従う乱数を戻す。指数分布の確率変数は、0から無限大までを範囲としています。また、-Log(Random Uniform ())（一様乱数の対数の符号を変えたもの）と等価です。

Random F(dfn, dfd, <noncentral>)

説明

指定された *dfn*（分母の自由度）と *dfd*（分子の自由度）のF分布に従う乱数を戻す。非心度パラメータの値はゼロ（デフォルト値）以上でなければなりません。

Random Frechet(<mu>, <sigma>)

説明

位置 *mu*、尺度 *sigma* のFréchet分布に従う乱数を戻す。デフォルト値は *mu* = 0 および *sigma* = 1 です。

Random Gamma(lambda, <scale>)

説明

形状パラメータ *lambda* およびオプションの尺度パラメータ *scale* のガンマ分布に従う乱数を生成する。尺度パラメータ *scale* のデフォルト値は1です。

Random Gamma Poisson(lambda, <sigma>)

説明

パラメータ *lambda* と *sigma* のガンマPoisson分布に従う乱数を生成する。*sigma* のデフォルト値は1です。

Random GenGamma(<mu=0>, <sigma=1>, <lambda=0>)

説明

パラメータ *mu*、*sigma*、*lambda* の拡張一般化ガンマ分布に従う乱数を生成する。

Random Geometric(p)

説明

1回の試行における成功確率が *p* である幾何分布に従う乱数を生成する。生成された値は、成功するまでの失敗回数。

Random GLog(mu, sigma, lambda)

説明

一般化対数分布に従う乱数を生成する。

Random Index(n, k)

説明

1 ～ *n* までの重複しない整数の乱数を含む *k* x 1 行列を戻す。

Random Integer(k, n)

説明

1 ～ *n* または *k* ～ *n* までの整数の乱数を生成する。

Random Johnson Sb(gamma, delta, theta, sigma)

説明

Johnson Sb 分布に従う乱数を生成する。

Random Johnson Sl(gamma, delta, theta, <sigma>)

説明

Johnson Sl 分布に従う乱数を生成する。sigma のデフォルト値は 1 です。

Random Johnson Su(gamma, delta, theta, sigma)

説明

Johnson Su 分布に従う乱数を生成する。

Random LEV(<mu>, <sigma>)

説明

位置 *mu*、尺度 *sigma* の最大極値分布に従う乱数を生成する。デフォルト値は *mu* = 0 および *sigma* = 1 です。

Random LogGenGamma(<mu=0>, <sigma=1>, <lambda=0>)

説明

パラメータ *mu*、*sigma*、*lambda* の対数一般化ガンマ分布に従う乱数を生成する。

Random Logistic(<mu>, <sigma>)

説明

位置 *mu*、尺度 *sigma* のロジスティック分布に従う乱数を生成する。デフォルト値は *mu* = 0 および *sigma* = 1 です。

Random Loglogistic(<mu>, <sigma>)

説明

位置 *mu*、尺度 *sigma* の対数ロジスティック分布に従う乱数を生成する。デフォルト値は *mu* = 0 および *sigma* = 1 です。

Random Lognormal(<mu>, <sigma>)

説明

位置パラメータ *mu*、尺度パラメータ *sigma* の対数正規分布に従う乱数を生成する。デフォルト値は *mu* = 0 および *sigma* = 1 です。

Random Negative Binomial(r, p)

説明

成功確率が *p*、成功回数が *r* の負の二項分布に従う乱数を生成する。生成された乱数は、r 回成功するまでに失敗した回数。

Random Normal(<mu>, <sigma>)

説明

平均が *mu*、標準偏差が *sigma* の正規分布に従う乱数を生成する。デフォルト値は *mu* = 0 および *sigma* = 1 です。

正規分布は、西洋の教会にあるベルの形状をした、左右対称な分布です。

Random Normal Mixture(mean, std_dev, probabilities)

説明

引数で指定された正規混合分布に従う乱数を生成する。

引数

mean グループ平均を示すベクトル
std_dev グループ標準偏差を示すベクトル
probabilities グループ確率を示すベクトル

Random Poisson(lambda)

説明

パラメータ *lambda* のポアソン分布に従う乱数を生成する。

Random Reset(seed)

説明

指定されたシード値（*seed*；初期値）から乱数系列を再開する。

Random SEV(<mu>, <sigma>)

説明

位置 *mu*、尺度 *sigma* の最小極値分布に従う乱数を生成する。デフォルト値は *mu* = 0 および *sigma* = 1 です。

Random Seed State(<seed state)

説明

BLOB オブジェクトの乱数シード値の状態を読み込むか、または設定する。

Random Shuffle(matrix)

説明

要素をランダムにシャッフルした行列を戻す。

Random t(df, <noncentral>)

説明

指定された *df*（自由度）の t 分布に従う乱数を生成する。非心度引数の値は負でも正でもかまいません。*noncentral* のデフォルト値は 0 です。

Random Triangular(min, mode, max)

Random Triangular(mode, max)

Random Triangular(mode)

説明

0 ～ 1 の値の三角分布に従う乱数を生成する（最頻値は *mode* で指定する）。三角分布は、通常、データ数の少ない母集団で使用されます。

引数

`min` 三角分布の下限値を指定する。デフォルト値は 0 です。

`mode` 三角分布の最頻値を指定する。

`max` 三角分布の上限値を指定する。デフォルト値は 1 です。

ノート

最頻値のみを指定した場合は、最小値は 0、最大値は 1 になります。最頻値と最大値を指定した場合は、最小値はデフォルトで 0 となります。

Random Uniform(<x>)

Random Uniform(<min>, <max>)

説明

0 ～ 1 の値の一様分布に従う乱数を生成する。Random Uniform(x) では 0 ～ x の間で乱数が生成されます。Random Uniform (min, max) では *min* ～ *max* の間で乱数が生成されます。生成された乱数データは、ほぼ均等に分布します。

Random Weibull(beta, <alpha>)

説明

Weibull 分布に従う乱数を戻す。*alpha* のデフォルト値は 1 です。

Resample Freq(<rate, <column>>)

説明

復元抽出法（重複抽出法）に基づき度数の列を生成する。引数が指定されていない場合は、元データと同じ標本サイズの標本が抽出されます。

引数

rate（オプション）標本の再抽出率。デフォルト値は 1 です。

column（オプション）列を指定した場合は、抽出率も指定する必要がある。標本サイズは、指定された列の和に抽出率を掛けたものとなります。

ノート

この関数を使用すると、0、1、2 といった度数を含む列が生成されます（通常、度数が 1 である行が多い）。これらの度数は、元の標本から無作為に *n* 行選択することにより作成されます。

また、**既存の**度数列に対してこの関数を使用すると、元の度数列とほぼ同じ値（期待値は同じ値）の列が生成されます。（既存の度数列に比例する割合で）ランダムに選択するため、既存の度数列の値とは多少異なります。

例

スクリプトを実行するたびに、度数列が同じ数値になるようにするには、As Constant() を使用します。As Constant() では、いったん式を評価して定数を求めると、その値は変化しません。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dc = dt << New Column( "column",
   Formula(
      As Constant(
         Random Reset( 123 );
         0;
      ) + Resample Freq()
   )
);
dc << Eval Formula;
```

行関数

As Table(matrix, <matrix 2, ...>, < <<invisible >, < <<Column Names(list) >)

説明

行列 *matrix* から新しいデータテーブルを作成する。

戻り値

新しいデータテーブル

引数

matrix 任意の行列

<<invisible 非表示のデータテーブルを作成する。データテーブルは、「JMP ホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。

<<Column Names(list) データの列名を指定したリスト。引数は引用符付きの列名のリストです。

Col Stored Value(<dt>, col, <row>)

説明

列プロパティ（「欠測値のコード」など）を使って割り当てられた値ではなく、実際に列に保存されているデータ値を戻す。

引数

dt（オプション）データテーブルへの参照。この値が指定されていない場合は、現在のデータテーブルが使用される。

col 列の名前。

row（オプション）行の名前または番号。この値が指定されていない場合は、現在の行が使用される。

例

x1 列に「欠測値のコード」列プロパティが割り当てられており、「999」が欠測値として扱われているとします。もう 1 つの列には平均を計算する計算式が含まれています。平均を計算するときに、欠測値ではなく「999」を使用するためには、式の中に Col Stored Value() を使用します。

```
Mean( Col Stored Value( :x1 ), :x2, :x3 )
```

Column(<dt>, "name", "formatted")

Column(<dt>, n)

説明

データテーブル列への参照を取得する。

引数

dt（オプション）データテーブルへの参照。指定されていない場合は、現在のデータテーブルが使用される

name 列名を指定する引用符付き文字列

formatted 設定されている表示形式でセルの値を戻すよう指定する引用符付き文字列。

n 列番号

Column Name(n)

説明

n で指定された列の名前を取得する。

戻り値

第 *n* 列の名前を（文字列ではなく）式として戻す

引数

n 列番号

Count(from, to, step, times)

説明

列計算式で使用される。*from* から *to* までインクリメントする値を各行に割り当てます。*steps* には、*from* と *to* も含めた、from から to までの値の個数を指定します。count 関数の最初の 3 つの引数によって決められた等差数列の値が、指定された *times* の数だけ繰り返して作成されます。*to* 値に達すると、count は再び *from* 値から開始されます。*from* と *to* の引数がデータテーブル列名のときは、Count は最初の行の値を取り、それ以降の行の値は無視されます。

戻り値

最後の値

引数

from 数値、列参照、または式。Count はこの値からカウントを開始する

to 数値、列参照、または式。Count はこの値でカウントを終了する

step 数値または式。*from* から *to* まで（両者を含む）のカウントに使用するステップの数を指定する

times 数値または式。次のステップに進むまでに同じ値を繰り返す回数を指定する

例

```
For Each Row(:colname[row()]=count(0, 6, 3, 1))
//colname という名前の列は、0,3,6,0, ... という数列ですべての行が埋められる

For Each Row(:colname[row()]=count(0, 6, 3, 2))
//colname という名前の列は、0,0,3,3,6,6,0, ... という数列ですべての行が埋められる
```

ノート

Count() は Row() に依存するので、主に列の計算式内で利用すると便利です。

Current Data Table(<dt>)

説明

引数が指定されていない場合、現在（最前面）のデータテーブルを取得する。引数が指定されている場合は、それを現在のデータテーブルとします。

戻り値

現在のデータテーブルへの参照

引数

dt（オプション）データテーブルの名前またはデータテーブルへの参照

Data Table(n)

Data Table("name")

説明

開いている *n* 番目のデータテーブル、または、引用符付き文字列で指定された名前（*name*）のデータテーブルへの参照を戻す。

戻り値

指定のデータテーブルへの参照

引数

n データテーブルの番号

name データテーブル名を指定する引用符付き文字列

Dif(col, n)

説明

列 *col* において、現在行の値から、現在行の *n* 行前の値を引いた差を計算する。

戻り値

差

引数

col 列名（たとえば :age など）

n 数値

Lag(col, n)

説明

n 行前の列 (col) の値を戻す。

N Row(dt); NRow(matrix)

N Rows(dt); NRows(matrix)

説明

指定されたデータテーブル（*dt*）や行列（*matrix*）の行数を戻す。

N Table()

説明

開いているデータテーブルの数を戻す。

New Column("name", attributes)

説明

データテーブル（*dt*）の最後に、指定された名前（"*name*"）で新しい列を追加する。

ノート

メッセージとしても使用可能です。dt<<New Column("name", attributes)

New Table("name", <visibility("invisible" | "private" | "visible")>, <actions>)

説明

指定された名前（*name*）で新しいデータテーブルを作成する。

引数

name 新しいテーブル名を示す引用符付き文字列。

visibility（オプション）引用符付きキーワード。invisible を指定すると、データテーブルは、「JMP ホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。private を指定すると、データテーブルを非表示で開き、JMP ホームウィンドウにも［ウィンドウ］メニューにもテーブル名を表示しません。visible を指定すると、データテーブルは表示されます。"visible" がデフォルト値です。

actions（オプション）新しいデータテーブルを定義するための引数。

Row()

Row() = y

説明

現在の行の番号を戻すか、設定する。引数はとりません。

Sequence(from, to, <stepSize>, <repeatTimes>)

説明

データテーブルの行に連続した数値を挿入する。*stepSize* および *repeatTimes* 引数はオプションで、どちらもデフォルト値は 1 です。

Subscript(a, b, c)

list[i]

matrix[b, c]

説明

リストや行列に対して、添え字を指定する。リスト（*list*）からは、*i* 番目の項目が抽出されます。または、行列（*matrix*）からは、第 *b* 行、第 *c* 列の要素が抽出されます。

Suppress Formula Eval(Boolean)

説明

引数が 1 の場合、すべてのデータテーブルの計算式の自動評価をオフにする。引数が 0 の場合は、オンにする。

行の属性関数

As Row State(i)

説明

整数 *i* を、行属性に変換する。

戻り値

指定された整数 *i* に対応した行属性

引数

i 整数

Color Of(rowstate)

説明

色番号を戻すか、設定する。

戻り値

行の属性（*rowstate*）の色番号

引数

rowstate 行の属性

例

5 行目に赤を割り当てます。

```
Color Of(Row State(5))=3
```

Color State(i)

説明

色番号を *i* とする行属性を戻す。

戻り値

行の属性

引数

i JMP の色番号

Combine States(rowstate, rowstate, ...)

説明
複数の行属性を組み合わせる。

戻り値
組み合わせた行属性

引数
rowstate 2つ以上の行属性

Excluded(rowstate)

説明
「除外する / 除外しない」の状態 (1 または 0) を戻すか、もしくは設定する。

戻り値
除外の属性（0 または 1）

引数
rowstate 1つ以上の行属性

Excluded State(num)

説明
除外の状態を指定された数値（*num*）に設定した行属性を戻す。

Hidden(rowstate)

説明
「表示しない / 再表示」の状態 (1 または 0) を戻す、もしくは、設定する。

Hidden State(num)

説明
非表示の状態を指定された数値（*num*）に設定した行属性を戻す。

Hue State(num)

説明
色相の状態を指定された数値（*num*）に設定した行属性を戻す。

Labeled(rowstate)

説明
「ラベルあり / ラベルなし」の状態 (1 または 0) を戻す、もしくは設定する。

Labeled State(num)

説明

ラベルの状態を指定された数値（*num*）に設定した行属性を戻す。

Marker Of(rowstate)

説明

行の属性のマーカーインデックスを戻すか、設定する。

Marker State(num)

説明

マーカーの状態を指定された数値（*num*）に設定した行属性を戻す。

Row State(<dt,> <n>)

説明

アクティブな行または *n* 行目において、初期値と異なる状態にある行属性を戻す。

引数

dt （オプション）位置指定引数。データテーブルへの参照。この引数が割り当ての形式をとらない場合は、データテーブルの式とみなされます。

n 行番号。

例

次の例は、データテーブル参照を作成し、「Big Class.jmp」の 1 行目の行の属性を戻します。

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt2 = Open( "$SAMPLE_DATA/San Francisco Crime.jmp" );
Row State( dt1, 1 );
```

Selected(rowstate)

説明

選択されている状態なのか、選択されていない状態なのか (1 または 0) を戻す、もしくは設定する。

Selected State(num)

説明

選択の状態を指定された数値（*num*）に設定した行属性を戻す。

Shade State(num)

説明

濃淡の状態を指定された数値（num）に設定した行属性を戻す。

SASインテグレーション関数

As SAS Expr(x)

説明

式を、SASのDATAステップで使用できる形式に変換し、文字列で戻す。コードは、PROC DS2のコールでラップする必要があります。リテラルな式を指定する場合には、Expr(...)で、その式を囲んでください。また、式が変数name内に格納されている場合には、NameExpr(name)と指定してください。このように指定しないと、式が評価された後の結果が渡されます。

戻り値

文字列

Current Metadata Connection()

説明

アクティブなSASメタデータサーバー接続があれば、それをスクリプト可能なオブジェクトとして取得する。

Current SAS Connection()

説明

アクティブなグローバルSASサーバー接続があれば、それをスクリプト可能なオブジェクトとして取得する。

Get SAS Version Preference()

説明

「環境設定」の「SASインテグレーション」ページで指定されているSASバージョンを戻す。

JMP6 SAS Compatibility Mode(Boolean)

説明

1に設定すると、SASの演算子がJMPバージョン6の機能と互換性のあるモードで機能する。

```
Meta Connect(<"machine", port>, <"authDomain">, <"username">, <"password">,
<named arguments>)
Meta Connect(<Profile("profile name")>, <Password("password")>, <named
arguments>)
Meta Connect(<Environment("environment name")>, <named arguments>)
```

説明

SAS Metadata Server に接続する。引数が何も指定されていない場合、空の接続ウィンドウが表示されます。一部の引数が指定されている場合、その値が挿入されたウィンドウが表示されます。引数がすべて指定されている場合、接続が確立され、ウィンドウは表示されません。

戻り値

接続に成功したときは 1、そうでなければ 0

引数

`machine`（オプション）コンピュータの DNS 名を示す引用符付き文字列

`port` コンピュータ名（machine）を指定した場合は必ず指定する。メタサーバーが待機しているポートを示す引用符付き文字列または整数。

`authDomain`（オプション）ログイン情報の認証ドメインを示す引用符付き文字列。ユーザ（*username*）とパスワード（*password*）を指定しない場合は不要

`username`（オプション）接続のためのユーザ名を示す引用符付き文字列

`password`（オプション）接続のためのパスワードを示す引用符付き文字列

名前付き引数

名前付き引数はすべてオプションです。

`Profile("profile_name")` メタデータサーバー接続プロファイルの名前を示す引用符付き文字列。ここから接続情報が取得されます。

`Environment("environment_name")` WIP 環境の名前を示す引用符付き文字列。ここから接続情報が取得されます。

`Password("password")` 指定されたプロファイル名のパスワードを示す引用符付き文字列

`CheckPreferenceOnly(0|1)` これが指定されている場合、`Meta Connect` は、JMP 環境設定の「SAS インテグレーション」ページの［**SAS Metadata Server に接続する**］オプションの状態を戻す。このオプションがオンの場合、**Meta Connect** は 1 を戻し、オフの場合は 0 を戻します。

`Repository("string")` 接続先のリポジトリの名前を示す引用符付き文字列を引数にとる。

`ProfileLookup(0|1)` プロファイル名ではなくコンピュータ名（*machine*）とポート名（*port*）が指定され、かつ `ProfileLookup` が指定されている場合、指定された machine と port を使用しているメタデータサーバー接続プロファイルを探そうとする。そのようなプロファイルが見つかった場合は、そこからその他の接続情報（認証ドメイン、ユーザ名、パスワードなど）が取得されます。

`Prompt(Always|Never|IfNeeded)`（オプション）接続しようとするたびにプロンプトを表示させる場合は `Always`、プロンプトを表示せずに失敗させる場合は `Never`、与えられた引数での接続に失敗したときにプロンプトを表示させる場合は `IfNeeded`（デフォルト）のキーワードをとる。

`SASVersion("<version number>" <,Strict>)` メタデータサーバ接続の確立前に、「環境設定」で指定されている SAS バージョンを、version_number 引数で指定した値に変更しようと試みる。SAS バージョン番号が、引数に指定したものとは異なる番号にすでに固定されている場合、

SASVersion 引数の指定は失敗となります。デフォルトでは、SAS バージョンを設定できない場合も、メタデータサーバー接続の確立が試行されますが、第 2 引数として Strict を指定すると、SAS バージョンを変更できない事態がエラーとみなされ、JSL の処理はその時点で停止します。Strict を指定しない場合、SASVersion 引数はヒントとして扱われ、可能な場合は環境設定のバージョン番号が設定されます。バージョン番号を設定できない場合でも、接続は試行されます。引数を指定する順序によって、結果が異なる場合があります。SAS バージョンの変更は、SASVersion 引数が指定された直後に試行されます。このため、他の引数、特に MetaConnect の効力に影響が及ぶ場合があります。SASVersion の有効な値は、"9.1.3"、"9.2"、"9.3"、および "9.4" です。メモ : SASVersion 引数の使用は、［SAS インテグレーション］の環境設定で SAS サーバーバージョンを変更するのと同じ働きをします。

ノート

引数が指定されておらず、プロファイルが保存されていない場合は、「SAS Metadata Server に接続する」ウィンドウが表示されます。

Meta Create Profile("profile", <named arguments>)

説明

メタデータサーバー接続プロファイルを作成し、それを既存のプロファイルの一覧に保存する。

戻り値

プロファイル（*profile*）が正常に作成されたときは 1、そうでなければ 0

引数

profile 作成されたプロファイルの名前を示す引用符付き文字列。指定した名前のプロファイルがすでに存在する場合、*Replace* が指定されていない限り、MetaCreateProfile は失敗します。

名前付き引数

以下の名前付き引数はすべてオプションです。

HostName("name") このプロファイルの接続先のホストコンピュータ名を示す引用符付き文字列。接続したい SAS Metadata Server が実行されているコンピュータの名前を指定してください。

Port(n) SAS Metadata Server が接続のためにリスンするポート番号（*n*）

AuthenticationDomain("domain") | AuthDomain("domain") 接続に使用する認証ドメインを設定する引用符付き文字列

Description("desc") | Desc("desc") このプロファイルの説明を設定する引用符付き文字列

Password("password") このプロファイルに保存するためのパスワードを示す引用符付き文字列

Replace(0|1) 名前（*name*）がすでに存在するプロファイルと一致した場合、既存のプロファイルを指定したプロファイルで置き換えるには、Replace が指定されていなければならない。デフォルト値は 0（偽）。

Repository("repository") このプロファイルの接続先の SAS Metadata Server のリポジトリ名を示す引用符付き文字列。このオプションは、SAS 9.1.3 Metadata Server への接続にのみ有効です。

UserName("username") このプロファイルで SAS Metadata Server に接続するときのユーザ名を示す引用符付き文字列

UseSingleSignOn(0|1) これを指定した場合、このプロファイルは、シングルサインオン（現在は、統合 Windows 認証とも呼ばれている）を使用して SAS Metadata Server に接続しようとする。こ

のオプションは、SAS 9.2 以降の Metadata Server への接続にのみ有効です。UseSingleSignOn を真（1）に指定した場合、*UserName* と *Password* は指定できません。デフォルト値は 0（偽）。

Meta Delete Profile("name")

説明

現在、保存されているプロファイルの一覧から、指定されたメタデータサーバー接続プロファイルを削除する。

戻り値

プロファイルが正常に削除されたときは 1、そうでなければ 0

引数

name 削除するプロファイルの名前を示す引用符付き文字列

Meta Disconnect()

説明

現在 SAS Metadata Server への接続があれば、それを切断する。

戻り値

なし

Meta Get Repositories()

説明

現在の SAS Metadata Server への接続で使用できるリポジトリのリストを取得する。

戻り値

文字列で示したリポジトリのリスト

Meta Get Servers()

説明

現在のセッションの接続先 SAS Metadata Repository に登録されている SAS サーバーのリストを取得する。

戻り値

文字列で示したサーバー名のリスト

Meta Get Stored Process("path")

説明

現在接続している SAS Metadata Repository からストアドプロセスオブジェクトを取得する。

戻り値

スクリプト可能なストアドプロセスオブジェクト

引数

path メタデータ内のストアドプロセスのパスを示す引用符付き文字列

Meta Is Connected()

説明

SAS Metadata Server への接続が現在存在するかどうかを特定する。

戻り値

接続が存在すれば 1、そうでなければ 0

引数

なし

Meta Set Repository("repositoryName")

説明

メタデータの検索に使用する SAS Metadata Repository を設定する。

戻り値

設定に成功したときは 1、そうでなければ 0

引数

repositoryName 現在のリポジトリに設定したいリポジトリの名前を示す引用符付き文字列

SAS Assign Lib Refs("libref", "path", <"engine">, <"engine options">)

説明

アクティブなグローバル SAS サーバー接続上において、SAS ライブラリ参照を割り当てる。

戻り値

成功したときは 1、そうでなければ 0

引数

libref 割り当てるライブラリの参照名（最大 8 文字）を示す引用符付き文字列

path SAS サーバー上の、割り当てられるライブラリへの完全パスを示す引用符付き文字列

engine（オプション）SAS サーバーがこのライブラリのメンバにアクセスする際に使用するエンジンを示す引用符付き文字列

engine options（オプション）使用するエンジンに必要なオプションを示す引用符付き文字列

SAS Connect(<"machine_name">, <"port">, <named_arguments>)

説明

ローカル、リモート、または論理 SAS サーバーに接続する。

戻り値

スクリプト可能な SAS サーバーオブジェクト

引数

machine_name（オプション）物理コンピュータ名、またはメタデータ定義された（論理）サーバー名を示す引用符付き文字列。物理コンピュータ名の場合はポートも指定する必要があります。サーバー名の場合はポートは**指定できません**。*name* と *port* のどちらも指定されず、JMP が Windows 上で

実行されている場合は、ローカルコンピュータ上のSASへの（COMを介した）接続が試みられ、すべての名前付き引数が無視されます。

port（オプション）引用符付き文字列または整数。*name*が物理コンピュータ名の場合、これがそのコンピュータ上で接続に使用されるポートとなります。*name*がメタデータ定義された（論理）サーバーの場合、*port*は指定できません。

名前付き引数

名前付き引数はすべてオプションです。

UserName("name") 接続に必要なユーザ名を示す引用符付き文字列

Password("password")（オプション）接続のためのパスワードを示す引用符付き文字列

ReplaceGlobalConnection(0|1) ブール値。デフォルト値は1（真）。このオプションが真で、SASサーバー接続が正常に行われた場合、アクティブな接続がこの新たな接続に置き換わり、グローバルなSAS接続に関するJSL関数を実行した場合、この接続が使われるようになります。このオプションが偽の場合、現在のグローバルなSAS接続は変更されません。その場合、新しく接続したサーバー接続を用いるには、接続した時に戻されたSASサーバーオブジェクトにメッセージを送る必要があります。

ShowDialog(0|1) ブール値。デフォルト値は0（偽）。この値が真の場合、（ReplaceGlobalConnection以外の）他の引数は無視され、「SASサーバー接続」ウィンドウが表示されます。JSLプログラマは、このオプションを用いて、「SASサーバー接続」ウィンドウを開かせることができます。

Prompt(Always|Never|IfNeeded) キーワード。Alwaysでは、接続を試みる前に常にプロンプトを表示します。Neverでは、接続の試みに失敗した場合でもプロンプトを表示しません（失敗しても、ただ単にログにエラーメッセージが送られるだけ）。IfNeeded（デフォルト値）では、与えられた引数で接続に失敗した場合（または与えられた認証情報では接続できない場合）にプロンプトを表示します。

ConnectLibraries(0|1) ブール値。「環境設定」の「SASインテグレーション」ページの［メタデータで定義されたSASライブラリに自動的に接続する］の設定がデフォルト値です。真（1）の場合、SASサーバーへの接続時に、メタデータで定義されたすべてのライブラリに接続します（処理に時間がかかる場合があります）。偽（0）の場合、メタデータ定義のライブラリは接続されません。特定のライブラリに後から接続するには、SAS Connect Libref グローバル関数またはSASサーバーオブジェクトのConnect Librefメッセージを使用します。

SASVersion("<version number>" <,Strict>) メタデータサーバ接続の確立前に、「環境設定」で指定されているSASバージョンを、version_number引数で指定した値に変更しようと試みる。SASバージョン番号が、引数に指定したものとは異なる番号にすでに固定されている場合、SASVersion引数の指定は失敗となります。デフォルトでは、SASバージョンを設定できない場合も、メタデータサーバー接続の確立が試行されますが、第2引数としてStrictを指定すると、SASバージョンを変更できない事態がエラーとみなされ、JSLの処理はその時点で停止します。Strict引数を指定しない場合は、SASVersion引数はヒントとして扱われ、可能な場合は環境設定のバージョン番号が設定されます。設定できない場合も、接続は試行されます。引数を指定する順序によって、結果が異なる場合があります。

SASバージョンの変更は、SASVersion引数が指定された直後に試行されます。

このため、他の引数、特に MetaConnect の効力に影響が及ぶ場合があります。SASVersion の有効な値は、"9.1.3"、"9.2"、"9.3"、および "9.4" です。メモ : SASVersion 引数の使用は、［SAS インテグレーション］の環境設定で SAS サーバーバージョンを変更するのと同じ働きをします。

SAS Deassign Lib Refs("libref")

説明

アクティブなグローバル SAS サーバー接続上において、SAS ライブラリ参照を解除する。

戻り値

成功したときは 1、そうでなければ 0

引数

libref 解除するライブラリの参照名を示した引用符付き文字列

SAS Disconnect()

説明

アクティブなグローバル SAS サーバー接続があれば、それを切断する。

戻り値

SAS 接続が確立されていて正常に切断できたときは 1、そうでなければ 0

引数

なし

SAS Export Data(dt, "library", "dataset", <named_arguments>)

説明

JMP データテーブルを、アクティブなグローバル SAS サーバー接続のライブラリ内の SAS データセットに書き出す。

戻り値

データテーブルが正常に書き出されたときは 1、そうでなければ 0

引数

dt データテーブルまたはデータテーブルへの参照
"library" データテーブルの書き出し先ライブラリ
"dataset" 新しい SAS データセットの名前

名前付き引数

名前付き引数はすべてオプションです。

Columns(list)|Columns(col1, col2, ...) 列のリスト、またはカンマで区切った列のリスト

Password("password") 書き出された SAS データセットの READ、WRITE、および ALTER のパスワードとする文字列。書き出されたデータセットが ALTER 用パスワードを使って現在のデータセットを上書きする場合、この ALTER 用パスワードが使用されます。*Password* を指定した場合、*ReadPassword*、*WritePassword*、および *AlterPassword* の値は無視されます。

ReadPassword("password") 書き出された SAS データセットの READ 用パスワードとする文字列。

`WritePassword("password")` 書き出されたSASデータセットのWRITE用パスワードとする文字列。

`AlterPassword("password")` 書き出されたSASデータセットのALTER用パスワードとする文字列。書き出されたデータセットがALTER用パスワードを使って現在のデータセットを上書きする場合、このALTER用パスワードが使用されます。

`PreserveSASColumnNames(0|1)` ブール値。これが1（真）で、**かつ**、JMPデータテーブルが元々SASデータセットであった場合、書き出されたSASデータセットにおいて、元のSASデータセットにおける列名が使用されます。デフォルト値は0（偽）。

`PreserveSASFormats(0|1)` ブール値。これが1（真）で、**かつ**、JMPデータテーブルが元々SASデータセットであった場合、書き出されたSASデータセットにおいて、元のSASデータセットにおけるフォーマットが使用されます。デフォルト値は1（真）。

`ReplaceExisting(0|1)` ブール値。1（真）の場合、指定のライブラリ内の指定された名前の既存のSASデータセットが、書き出されたSASデータセットに置き換わります。0（偽）の場合、指定のライブラリ内に指定された名前のデータセットがすでにあれば、書き出しは中止されます。デフォルト値は0（偽）。

`SaveJMPMetadata(0|1)` SAS 9.4拡張属性にJMPメタデータ（テーブルスクリプトと列のプロパティなど）を保存する。デフォルト値は0（偽）。

`HonorExcludedRows(0|1)` ブール値。1（真）の場合、JMPデータテーブルで除外されている行は書き出されません。デフォルト値は0（偽）。

ノート

書き出しに関する情報はログに送られます。

SAS Get Data Sets("libref")

説明

SASライブラリ内にあるデータセットのリストを戻す。

戻り値

文字列のリスト

引数

`libref` SASライブラリ参照名または表示ライブラリ名を示す引用符付き文字列。引数で指定されたライブラリ内にあるSASデータセットの名前が、リストで戻されます。

SAS Get File("source", "dest", "encoding")

説明

アクティブなグローバルSASサーバー接続からファイルを取得する。JMPは、FILENAMEステートメント（指定されている場合はエンコーディングを含む）を生成し、そのステートメントを使ってSASサーバー上のファイルを読み取ります。

戻り値

成功したときは1、そうでなければ0

引数

source クライアントコンピュータにダウンロードするサーバー上のファイルの完全パスを示した引用符付き文字列

dest サーバーからダウンロードしたファイルのコピーを格納するクライアントコンピュータ上の場所の完全パスを示した引用符付き文字列

encoding ファイルで使用されているエンコーディングを示す引用符付き文字列（"utf-8" など）。サーバーでサポートされているエンコーディングを指定する必要があります。

SAS Get File Names("fileref")

説明

アクティブなグローバル SAS サーバー接続から、与えられたファイル参照にあるファイル名のリストを取得する。

戻り値

文字列のリスト

引数

fileref ファイル参照名を示した引用符付き文字列。指定されたファイル参照の配下にあるファイルの名前が取得されます。

SAS Get File Names In Path("path")

説明

アクティブなグローバル SAS サーバー接続から、与えられたパスにあるファイル名のリストを取得する。

戻り値

文字列のリスト

引数

path ファイル名を取得するサーバー上のディレクトリパスを示した引用符付き文字列

SAS Get File Refs()

説明

アクティブなグローバル SAS サーバー接続から、現在定義されている SAS ファイル参照のリストを取得する。

戻り値

戻り値のリストは、2 つのリストで構成されています。最初のリストは、ファイル参照名を、引用符付き文字列で含んでいます。2 番目のリストは、物理的なパス名を、引用符付き文字列で含んでいます。

SAS Get Lib Refs(<named arguments>)

説明

現在のグローバル SAS サーバー接続から、現在定義されている SAS ライブラリ参照のリストを戻す。

戻り値

文字列のリスト

名前付き引数

Friendly Names(0|1)（オプション）ブール値。1（真）の場合、表示名を持つライブラリ（メタデータ定義のライブラリ）に関しては、8文字のライブラリ参照名ではなく、表示名が戻される。

SAS Get Log()

説明

現在のグローバルなSASサーバー接続から、SASログを取得する。

戻り値

文字列

SAS Get Output()

説明

現在のグローバルなSASサーバー接続へ最後に送信したSASコードの出力リストを戻す。

戻り値

文字列

SAS Get Results()

説明

前回のSASサブミットの結果をスクリプト可能なオブジェクトとして取得する。これにより、結果を柔軟に処理できるようになります。

戻り値

スクリプト可能なSAS実行結果オブジェクト

SAS Get Var Names(string, <"dataset">, <password("password")>)

説明

このSASサーバー接続から、指定のデータセットに含まれている変数の変数名を取得する。

戻り値

文字列のリスト

引数

string 以下のいずれかを示す引用符付き文字列

- 読み込むSASデータセットを示すSASライブラリの名前。この場合、*dataset*の名前引数が必要です。
- 読み込むSASデータセットの完全メンバ名。"ライブラリ名.メンバ名"で指定します。
- 読み込む論理SASデータテーブルへのSAS Foldersツリーのパス。このオプションを使用するには、SAS 9.2以降のMetadata Serverへの接続が必要です。

dataset（オプション）変数名を取得するデータセットの名前を示す引用符付き文字列

password("password")（オプション）データセットの読み取り用パスワードを示す引用符付き文字列。このオプションを指定しない場合、読み取り用パスワードが設定されたデータセットを開こうとすると、パスワードの入力を促すダイアログが表示されます。

SAS Import Data(string, <"dataset">, <named arguments>)

説明

アクティブなグローバルSASサーバー接続から、JMPデータテーブルにSASデータセットを読み込む。

戻り値

JMPデータテーブルオブジェクト

引数

`string` 以下のいずれかを示す引用符付き文字列

- 読み込むSASデータセットを示すSASライブラリの名前。この場合、*"dataset"*の名前引数が必要です。
- 読み込むSASデータセットの完全メンバ名。"ライブラリ名.メンバ名"で指定します。
- 読み込む論理SASデータテーブルへのSAS Foldersツリーのパス。このオプションを使用するには、SAS 9.2以降のMetadata Serverへの接続が必要です。

`dataset`（オプション）データセットの名前を示す引用符付き文字列

名前付き引数

名前付き引数はすべてオプションです。

`Columns("list")|Columns(col1, col2, ...)` 読み込みに含める列の名前を示す引用符付き文字列リストまたは複数の文字列

`ConvertCustomFormats(0|1)` デフォルト値は1（真）。この値が真で、かつ読み込まれるSASデータセット内にユーザ定義のフォーマットが検出された場合、それらのフォーマットをJMPの値ラベルに変換する試みが行われます。

`Invisible(0|1)` デフォルト値は0（偽）。1（真）の場合は、JMPデータテーブルが非表示となります。データテーブルは、「JMPホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。非表示のデータテーブルは、ユーザによって明示的に閉じられるまでメモリ内に保管され、その分JMPが使用できるメモリが減ることになります。非表示のテーブルを明示的に閉じるには、`Close(dt)` を実行します。ここで、*dt* は、SASImportData関数から戻されたデータテーブル参照の変数です。

`Where("filter")` `Where("salary<50000")` のように、データの読み込みに使用するフィルタを示す引用符付き文字列

`Password("password")` データセットの読み取り用パスワードを示す引用符付き文字列。このオプションを指定しない場合、読み取り用パスワードが設定されたデータセットを開こうとすると、パスワードの入力を促すダイアログが表示されます。

`UseLabelsForVarNames(0|1)` 1（真）の場合、SASデータセットのラベルを、JMPデータテーブルの列名にします。0（偽）の場合、SASデータセットの変数名を列名とします。デフォルト値は0（偽）。

`RestoreJMPMetadata(0|1)` SAS 9.4拡張属性にJMPメタデータを保存する。デフォルト値は0（偽）。

`Sample(named arguments)`（オプション）名前付き。SASデータセットから、無作為抽出して、JMPデータテーブルに読み込みます。WhereとSampleの両方を指定した場合は、まずWhere節によって一部分を抽出してから、Sampleで指定された方法で無作為抽出されます。Sampleを指定した場合、SASサーバーにおいて、PROC SURVEYSELECTが実行されます。このプロシジャを実行するには、SASサーバーにおいて、SAS/STATパッケージのライセンスがあり、インストールされていなければ

なりません。無作為抽出に関しては、SAS が提供しているドキュメントの PROC SURVEYSELECT に関する章を参照してください。デフォルト（引数が指定されない場合）では、抽出率が 5% の単純無作為抽出が行われます。Sample に使用できる引数は以下のとおりです（すべてオプション指定です）。

- Simple | Unrestricted: Simpleを指定した場合、不重複抽出（非復元抽出）が行われる。Unrestrictedを指定した場合、重複抽出（復元抽出）が行われます。これら2つのオプションは、お互いに排他的で、どちらか一方しか指定できません。
- SampleSize(int) | N(int): 標本の合計行数。層化抽出の場合は層ごとの行数。
- SampleRate(number) | Rate(number) | Percent(number): 標本抽出率を指定する。層化抽出の場合、標本抽出率は各層に適用されます。値はパーセンテージで指定します。SampleRate(3.5)は、標本抽出率3.5%を意味します。
- Strata({col1, col2, ...}) | Strata(col1, col2, ...): 層別変数として指定された列名で、層化無作為抽出を実行する。
- NMin(int): （全体の、または層化した標本の層ごとの）標本の最小行数。標本抽出率を指定した場合にのみ適用されます。
- NMax(int): （全体の、または層化した標本の層ごとの）標本の最大行数。標本抽出率を指定した場合にのみ適用されます。
- Seed(int): 標本のシード値として使用する数値。プログラムを実行した際に同じ標本が抽出されるようにしたい場合に便利です。デフォルトでは、シードは日時に基づいた乱数です。詳細については、SASが提供しているドキュメントのPROC SURVEYSELECTに関する章を参照してください。
- OutputHits(0|1): ブール値。デフォルト値は0（偽）。Unrestrictedを指定して重複抽出（復元抽出）を行う際、デフォルトでは、2回以上抽出された行は、結果のJMPデータテーブルにおいて1行しかなく、NumberHits列に、その行の抽出回数が示されます。OutputHitsを1（真）に設定すると、複数回、抽出された行は、結果のJMPデータテーブルにおいて複数行で示されます。
- SelectAll(0|1): ブール値。デフォルト値は1（真）。SelectAllを1（真）に設定すると、層の標本サイズが層内の行数を超えた場合にその層の行すべてが抽出されます。SelectAllを0（偽）に設定すると、層の標本サイズが層内の行数を超えた場合には、エラーが出力されます。SelectAllは、抽出法としてSimpleを指定した場合にのみ適用されます。

SQLTableVariable(0|1) このオプションが 1（真）の場合、結果の JMP データテーブルに、SQL 文を含んだテーブル変数が作成されます。この SQL 文は、データを取得するために SAS に送信されたものです。0（偽）の場合、SQL 文のテーブル変数は作成されません。デフォルト値は 1（真）。なお、読み取り用パスワードが必要なデータセットに対しては、作成された SQL 文において、パスワードは隠された状態になります。

SAS Import Query("sqlquery", <named arguments>)

説明

要求された SQL クエリーを現在のグローバルな SAS サーバー接続で実行し、その結果を JMP データテーブルに読み込む。

戻り値

JMP データテーブルオブジェクト

引数

`sqlquery` SQL クエリーを示した引用符付き文字列。この SQL クエリーが実行され、その結果が読み込まれます。

名前付き引数

名前付き引数はすべてオプションです。

`ConvertCustomFormats(0|1)` デフォルト値は 1（真）。この値が真で、かつ読み込まれる SAS データセット内にユーザ定義のフォーマットが検出された場合、それらのフォーマットを JMP の値ラベルに変換する試みが行われます。

`Invisible(0|1)` デフォルト値は 0（偽）。1（真）の場合は、JMP データテーブルが非表示となります。データテーブルは、「JMP ホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。非表示のデータテーブルは、ユーザによって明示的に閉じられるまでメモリ内に保管され、その分 JMP が使用できるメモリが減ることになります。非表示のテーブルを明示的に閉じるには、`Close(dt)` を実行します。ここで、*dt* は `SASImportData` 関数から戻されたデータテーブル参照の変数です。

`UseLabelsForVarNames(0|1)` デフォルト値は 1（真）。1（真）の場合、SAS データセットのラベルを、JMP データテーブルの列名にします。0（偽）の場合、SAS データセットの変数名を列名とします。

`RestoreJMPMetadata(0|1)` JMP メタデータの保存時に SAS 9.4 拡張属性を含めます。デフォルト値は 0（偽）。

`SQLTableVariable(0|1)` デフォルト値は 1（真）。このオプションが 1（真）の場合、結果の JMP データテーブルに、SQL 文を含んだテーブル変数が作成されます。この SQL 文は、データを取得するために SAS に送信されたものです。0（偽）の場合、SQL 文のテーブル変数は作成されません。なお、読み取り用パスワードが必要なデータセットに対しては、作成された SQL 文において、パスワードは隠された状態になります。

SAS Is Connected()

説明

アクティブなグローバル SAS サーバー接続があるかどうかを調べる。

戻り値

アクティブなグローバル SAS 接続があるときは 1、そうでなければ 0

SAS Is Local Server Available()

説明

ローカルの SAS サーバーが使用可能な場合は 1（真）を戻し、そうでない場合は 0（偽）を戻す。

SAS Load Text File("path")

説明

このSASサーバー接続から、パス（path）で指定されたファイルをダウンロードし、その内容を文字列で取得する。

戻り値

文字列

引数

"path" ダウンロードし、内容を文字列で取得するファイルの、サーバー上の完全パスを示した引用符付き文字列

SAS Name("name")

SAS Name({list of names})

説明

JMP変数名をSAS変数名に変換する。その際、特殊文字と空白を下線に変更するなどのさまざまな変換を行って有効なSAS名を生成します。

戻り値

スペースで区切った1つまたは複数の有効なSAS名を含む文字列

引数

"name" JMP変数名を示す引用符付き文字列、または引用符付きJMP変数名のリスト

SAS Open For Var Names("path")

説明

変数名の取得だけを目的としてSASデータセットを開き、変数名を文字列のリストで戻す。

戻り値

ファイル内の変数名のリスト

引数

path SASデータセットのパス名を示す引用符付き文字列

SAS Send File("source", "dest", "encoding")

説明

クライアントコンピュータから、アクティブなグローバルSASサーバー接続へファイルを送信する。JMPは、FILENAMEステートメント（指定されている場合はエンコーディングを含む）を生成し、そのステートメントを使ってSASサーバー上のファイルを読み取ります。

戻り値

成功したときは1、そうでなければ0

引数

source サーバーにアップロードするクライアントコンピュータ上のファイルの完全パスを示した引用符付き文字列

dest クライアントコンピュータからアップロードしたファイルをサーバー上のどの場所に格納するかを完全パスで指定する引用符付き文字列

encoding ファイルで使用されているエンコーディングを示す引用符付き文字列（"utf-8" など）。サーバーでサポートされているエンコーディングを指定する必要があります。

SAS Submit("sasCode", <named arguments>)

説明

アクティブなグローバル SAS サーバー接続に、SAS コードをサブミットする。

戻り値

成功したときは 1、そうでなければ 0

引数

sasCode サブミットする SAS コードを示す引用符付き文字列

名前付き引数

名前付き引数はすべてオプションです。

Async(0|1) ブール値。1（真）を指定した場合、非同期でバックグラウンドで、サブミットが行われます。サブミットの状態を判断するには、SAS Server スクリプト可能オブジェクトに、Get Submit Status() メッセージを送ります。デフォルト値は 0（偽）。

ConvertCustomFormats(0|1) ブール値。サブミットが完了し、サブミットされた SAS コードによって生成された SAS データセットが JMP に読み込まれた際（Open Output Datasets を参照）、SAS データセットの列にあるユーザ定義のフォーマットを JMP の値ラベルに変換する試みを行うかどうかを指定します。デフォルト値は 1（真）。

DeclareMacros(var1, var2, ...) JSL 変数の名前。ここで指定された JSL 変数の値が、マクロ変数として、SAS に渡されます。指定する各 JSL 変数は、文字列または数値を含むものでなければなりません。完全修飾された JSL 変数名は、変数名だけが SAS に送られます。たとえば、namespace:variable_name という JSL 変数名は、variable_name という SAS 変数名になります。

GetSASLog(<Boolean|OnError>, <JMPLog|Window>) ブール値。引数が指定されていない場合、SAS ログは、SAS インテグレーションの環境設定で指定されている場所に取得および表示されます。GetSASLog の第 1 引数は、ブール値またはキーワード OnError をとります。ブール値を指定した場合、1（真）は SAS ログを表示し、0（偽）は SAS ログを表示しません。OnError を指定した場合、サブミットでエラーが生じた場合にのみ SAS ログが表示されます。GetSASLog の第 2 引数は、SAS ログの表示場所を指定します。JMPLog を指定した場合、SAS ログは JMP ログに追加されます。Window を指定した場合、SAS ログは個別のウィンドウに表示されます。

GraphicsDevice(string) または GDevice(string) サブミットされた SAS コードによって生成されるグラフィックで、どの GDEVICE SAS オプションを使用するかを値で指定する引用符付き文字列。値は、有効な SAS グラフィックデバイスでなければなりません。デフォルト値は、環境設定で指定されている設定です。

Interactive(0|1) JMPは、生成されたラッパーコードにQUITステートメントを含めます。インタラクティブなプロシジャは、JMPがODSラッパーを生成している場合でも機能します。サブミットのたびに、インタラクティブシーケンスの一部である引数を指定します。指定しない場合、コードの先頭と末尾にQUITがつきます。

NoOutputWindow ブール値。1（真）の場合、SASの出力を表示する「SASアウトプット」ウィンドウが表示されません。デフォルト値は0（偽）。

ODS(0|1) 1（真）の場合、ODS結果を生成するための追加のSASコードが実行され、元のSASコードの結果が、ODS結果として表示されます。デフォルト値は、環境設定で指定されている設定です。

ODSFormat(string) 生成されたODS結果のフォーマットを指定する引用符付き文字列。有効な値は"HTML"、"RTF"、および"PDF"です。デフォルト値は、環境設定で指定されている設定です。

ODSGraphics(0|1) 1（真）の場合、サブミットされたSASコードのODS統計グラフィックが生成される。ODSGraphicsを真に設定すると、ODSも真に設定されます。デフォルト値は、環境設定で指定されている設定です。

ODSStyle(string) ODS結果の生成時にどのODSスタイルを使用するかを指定する引用符付き文字列。文字列（*string*）は有効なSASスタイルでなければなりません。デフォルト値は、環境設定で指定されている設定です。

ODSStyleSheet(path) 生成されたODS結果のフォーマット時にローカルのどのCSSスタイルシートを使用するかを指定する引用符付き文字列。*Path*は、（JMPを実行している）クライアントコンピュータにおいて有効な、CSSファイルのパス名を示すものでなければなりません。デフォルト値は、環境設定で指定されている設定です。

OnSubmitComplete(script) サブミットの完了時に実行するJSLスクリプトを指定する引用符付き文字列。これは非同期のサブミットに特に便利です。スクリプトが定義済みのJSL関数の名前である場合は、SAS Serverスクリプト可能オブジェクトを第1引数として、その関数が実行されます。

OpenODSResults(0|1) 1（真）の場合、サブミットが完了した後、サブミットされたSASコードによって生成されたODS結果が自動的に開く。デフォルト値は1（真）。

OpenOutputDatasets(<All|None|dataset1, dataset2, ...>) サブミットされたSASコードによって新しいSASデータセットが作成された場合、JMPによって検知されます。

OpenOutputDatasets（略称のOutDataも使用可）は、SASサブミットが完了した後、それらのデータセットをどのように扱うかを決定します。Allを指定した場合、SASコードによって生成されたすべてのデータセットは、SASサブミットの完了後、JMPに読み込まれます。Noneを指定した場合、生成されたデータセットはどれも読み込まれません。特定のデータセットだけが読み込まれるようにするには、それらのデータセット名を指定します。デフォルト値は、環境設定で指定されている設定です。

Title(string) ウィンドウタイトルを示す引用符付き文字列。指定された文字列は、サブミットしたODSの結果を表示するウィンドウのタイトルに使用される。

SAS Submit File("filename", <named arguments>)

説明

アクティブなグローバルSASサーバー接続で、ファイルに含まれたSASコードをサブミットする。

戻り値

成功したときは1、そうでなければ0

引数

`filename` 送信するSASコードを含むファイルの名前を示す引用符付き文字列

名前付き引数

SAS Submitと同様。

SQL関数

メモ: 文字`$# -+/%()&|;?`を含んでいるデータテーブル名 はサポートされません。

As SQL Expr(x)

説明

式を、SQLのSelectステートメントで使用できるコードに変換し、文字列で戻す。リテラルな式を指定する場合には、`Expr(...)`で、その式を囲んでください。また、式が変数name内に格納されている場合には、`NameExpr(name)`と指定してください。このように指定しないと、式が評価された後の結果が渡されます。

戻り値

文字列

New SQL Query(Connection ("ODBC:connection_string")|("SAS:connection_string"), Select(Column("column", "t1")), From(Table("table", <Schema("schema")>, <Alias("t1")>)), <Options(JMP 12 Compatible(1)|Run on Open(1)))>

New SQL Query(Connection("ODBC:connection_string;")|("SAS:connection_string;"), Custom("SELECT col1, col2, col3 FROM table;")), <Options(JMP 12 Compatible(1)|Run on Open(1))>

説明

指定した接続、列、データテーブル、または独自のSQLクエリーのSQLクエリーオブジェクトを作成する。

戻り値

クエリー対象のデータを含むデータテーブル。そのデータテーブルには、クエリーを変更および更新するためのSQLクエリー文字列とテーブルスクリプトが含まれています。

引数

`Connection` ODBC接続またはSAS接続のための文字列。

`Select` 選択したい列とその別名。

`From` クエリー対象のテーブル、および（オプションで）スキーマと列の別名。

`Custom` 指定したテーブルの列を選択するSQLステートメント。

Version クエリーを開くのに必要なJMPの最低バージョン。この条件が満たされない場合は、互換性に関するメッセージがログに書き込まれ、クエリーは開かれません。

Options ブール値。［クエリービルダー］の環境設定で、JMP 12との互換性を維持するオプションを選択すると（または、［クエリービルダー］の赤い三角ボタンのメニューで同等のオプションを選択すると）、生成されたスクリプトに JMP 12 Compatible が挿入されます。このオプションを指定すると、互換性に問題があるJMP 13のクエリーも、JMP 12で実行することができます。クエリーを編集モードで開くのではなく、開くと同時に実行するには、Run on Open(1) を指定します。

Query(<<dt1|Table(dt1, alias1)>, ..., <dtN, aliasN)>>, <private | invisible>, <scalar>, sqlStatement)

説明

選択されたデータテーブルに対し、SQLクエリーを実行する。

戻り値

クエリーの結果（データテーブルまたは1つの値）。

引数

dt1, dtN（オプション）データテーブルに割り当てられた変数。

Table（オプション）データテーブルへの参照を渡す。

alias1, aliasN データベーステーブルの別名を指定する。

private（オプション）結果のデータテーブルを表示しない。

invisible（オプション）テーブルを非表示にする。データテーブルは、「JMP ホームウィンドウ」と［ウィンドウ］メニューにのみ表示されます。非表示のデータテーブルは、ユーザによって明示的に閉じられるまでメモリ内に保管され、その分JMPが使用できるメモリが減ることになります。非表示のテーブルを明示的に閉じるには、Close(dt) を実行します。ここで、*dt* は、データテーブル参照の変数です。

scalar（オプション）クエリーが1つの値を戻すことを示します。

sqlStatement 必須。SQLステートメント（通常はSELECTステートメント）。SQLステートメントは、必ず、最後の引数として指定します。

例

次の例では、15歳以上の生徒のデータをすべて選択します。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
result = Query( Table( dt, "t1" ), "SELECT * FROM t1 WHERE 年齢 > 14;" );
```

ノート

Query() がサポートするSQLiteのコマンドについては、付録A「JMPクエリーで使用可能なSQL関数」を参照してください。その他の例については、『スクリプトガイド』の「JMPの拡張」章を参照してください。

統計関数

ARIMA Forecast(column, length, model, estimates, from, to)

説明

指定のモデルと予測値を使って、指定の列にある指定の行の予測値を戻す。

戻り値

引数 *from* と *to* によって指定された範囲の、*column* 列に対する予測値のベクトル

引数

column データテーブルの列

Length 使用する列内の行数

model 時系列モデルオプションのメッセージ

estimates 予測に用いるモデルの係数を表す名前付き値のリスト。時系列プラットフォームにて、ARIMA モデルをあてはめて、予測値を保存したときにも、このリストは生成されます。

from, to 値の範囲 通常、*from* には、1 以上から *to* 以下の整数のいずれかを指定します。*from* が 0 以下かつ *to* 以下の場合、結果は、実測値に対する予測値になります。

Best Partition(xindices, yindices, <<Ordered, <<Continuous Y, <<Continuous X)

説明

最適なグループ分けを探す関数。試験的な関数。

戻り値

リスト

引数

xindices, yindices 同次元の行列

Col Cumulative Sum(name, <By var, ...>)

Cumulative Sum(name)

説明

現在の行までの累積和を戻す。Col Cumulative Sum は、By 列をサポートしていますが、事前に By 列で並べ替えをしておく必要はありません。

引数

name 列名

By var オプション : グループごとに統計量を計算するには By 変数を指定する。

Col Maximum(name, <By var, ...>)

Col Max(name)

説明

指定された列の全行における最大値を計算する。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の最大値

引数

name 列名

By var オプション：グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Mean(name, <By var, ...>)

説明

指定された列の全行における平均値を計算する。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の平均値

引数

name 列名

By var オプション：グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Minimum(name, <By var, ...>)

Col Min(name)

説明

指定された列の全行における最小値を計算する。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の最小値

引数

name 列名

By var オプション : グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Moving Average(name, options, <By var, ...>)

Moving Average(name, options)

説明

指定された期間で、現在の行における移動平均を戻す。Col Moving Average は By 列をサポートしています。

引数

name 列名

Weighting(1|0|n) 位置引数。値への重みの付け方を指定する。1 の場合、すべての項に等しい重みを加える。0 の場合、線形に増加する重みを加える。その他の値の場合は、その値を指数加重移動平均のパラメータとして使用する（EWMA または EMA）。

Before(1|0|n) 位置引数。現在の項のいくつ前からの項を平均の範囲（ウィンドウ）に含めるかを指定する（現在の項を数に入れて）。-1 の場合は、後のすべての項を使用する。

After(1|0|n) 位置引数。現在の項のいくつ後までの項を平均の範囲（ウィンドウ）に含めるかを指定する（現在の項を数に入れて）。-1 の場合は、過去のすべての項を使用する。

Partial Window is Missing 位置引数（ブール値）。欠測値の扱いを指定する。デフォルトでは、欠測値は無視されます。

By var オプション : グループごとに統計量を計算するには By 変数を指定する。

例

```
Col Moving Average( x, 1, 4 );
// 5 つの項の移動平均を、等しい重みで求める
Col Moving Average( x, 0 );
// 過去のすべての項の移動平均を、線形に増加する重みを加えて求める
Col Moving Average( x, 0, 2, 2 );
// 現在の項に前後の 2 項つを含む 5 項目の三角移動平均を求める
Col Moving Average( x, 0.25 );
// 過去のすべての項の指数移動平均を求める
```

Col N Missing(name, <By var, ...>)

説明

指定された列の全行における欠測値の個数を求める。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の欠測値の個数

引数

name 列名

By var オプション：グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Number(name, <By var, ...>)

説明

指定された列の全行における非欠測値の個数を求める。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の非欠測値の個数

引数

name 列名

By var オプション：グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Quantile(name, p, <ByVar>)

説明

指定された列の行全体における分位点 *p* を求める。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の分位点

引数

name 列名

p 指定の分位点 *p*。0 ～ 1 の範囲で指定します。

ByVar By グループ

例

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Col Quantile( :Name("身長 (インチ)"), .5 );
```

63

戻り値の「63」は、「身長 (インチ)」列の 50% 点、つまり中央値（メディアン）です。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Rank(column, <ByVar, ...>, <<tie("average"|"arbitrary"|"row"|"minimum")

説明

最小値を 1 位、最大値を一番最後の順位として、各行に順位をつける。同順位のデータ値には、恣意的な順位が与えられます。

引数

ByVar（オプション）グループごとに統計量を計算するには By 変数を指定する。

column 順位付けされる列

<<tie 同じ値が複数ある場合、順位の付け方を決定する。[33 55 77 55] というデータの場合、33 が 1 位、77 が 4 位となり、2 つの 55 については、順位が定まらない。average を指定すると、両方とも、平均順位の 2.5 位になる。arbitrary を指定すると、2 位と 3 位を任意に割り当てる（JMP 12 ではこの方法で処理されていた）。row を指定すると、元のデータの順番に従う （1 つ目の 55 が 2 位、2 つ目の 55 が 3 位となる）。

minimum を指定すると、両方に上位の順位（2 位）を割り当てる。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Standardize(name)

説明

指定された列の全行を対象に、平均値を引いて標準偏差で割った値を算出する。

戻り値

標準化したデータ値

引数

name 列名

ノート

標準化とは、データから平均を引いて、それを標準偏差で割ることです。そのため、次の 2 つのコマンドは同じ結果になります。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << New Column( "stdht", Formula( Col Standardize(:Name("身長（インチ）")) ) );
dt << New Column( "stdht2",
   Formula( (:Name("身長（インチ）") - Col Mean( :Name("身長（インチ）") )) / Col Std
   Dev( :Name("身長（インチ）") ) )
);
```

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Std Dev(name,<By var, ...>)

説明

指定された列の全行における標準偏差を計算する。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の標準偏差

引数

name 列名

By var オプション：グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Col Sum(name,<By var, ...>)

説明

指定された列の全行における合計を計算する。全行が欠測値の場合、Col Sum 関数は欠測値を戻します。複数の評価を迅速に行えるよう、結果は内部にキャッシュされます。

戻り値

列の合計

引数

name 列名

By var オプション：グループごとに統計量を計算するには By 変数を指定する。

ノート

データ値が列プロパティ（「欠測値のコード」など）によって割り当てられている場合、代わりに列に保存されている値の計算を基にするには、Col Stored Value() を使用します。「Col Stored Value(<dt>, col, <row>)」（197 ページ）を参照してください。

Fit Censored(Distribution("name"), YLow(vector) | Y(Vector), <YHigh(vector)>, <Weight(vector)>, <X(matrix)>, <Z(matrix)>, <HoldParm(vector)>)

説明

打ち切りのあるデータに、指定された分布をあてはめる。

戻り値

パラメータ推定値、共分散行列、対数尤度、AICc、BIC、収束メッセージで構成されるリスト。

引数

`Distribution("name")` あてはめる分布の引用符付きの名前。

`YLow(vector) | Y(Vector)` 打ち切りがないデータの場合は、Y だけを指定し、YHigh は指定しない。打ち切りがあるデータの場合は、YLow および YHigh に、それぞれ、打ち切りの下限値と上限値を指定してください。

オプションの引数

`YHigh(vector)` 打ち切りの上限値を示すベクトル。打ち切りがある場合のみ、YLow と YHigh の 2 つを指定してください。

`Weight(vector)` 重み値を示すベクトル

`X(matrix)` 回帰モデルの位置に対する計画行列

`Z(matrix)` 回帰モデルの尺度に対する計画行列

`HoldParm(vector)` 固定するパラメータの配列。パラメータを固定する場合は非欠測値、自由パラメータとして推定する場合は欠測値を指定してください。このオプションは、「パラメータが、ゼロである」や「パラメータが、特定の値である」いう仮説に対する検定を、特定のパラメータに対して行いたいときに使ってください。

Hier Clust(x)

説明

データ行列 x について、Ward 法により（データを標準化せずに）階層型クラスター分析を行った履歴を戻す。

引数

`x` データ行列

IRT Ability(Q1, <Q2, Q3, ...,> parameter matrix)

説明

項目反応理論（IRT）モデルでの能力を求める。

KDE(vector, <named arguments>)

説明

バンド幅を自動選択して、カーネル密度推定値を戻す。

引数

`vector` ベクトル

名前付き引数

名前付き引数はすべてオプションです。

`<<weights` vector と同じ長さのベクトル。負でない任意の実数を含めることができます。度数や重みなどを指定するときに用います。

`<<bandwidth(n)` 負でない実数。0 を指定した場合、バンド幅は自動選択されます。

`<<bandwidth scale(n)` 正の実数。

<<bandwidth selection(n) バンド幅の自動選択方法として、0（Sheather and Jones）、1（正規分布参照）、2（Silverman の経験則）、3（過平滑化）のいずれかを指定してください。

<<kernel(n) カーネル関数として、0（Gauss）、1（Epanechnikov）、2（双加重）、3（三角）、4（矩形）のいずれかを指定してください。

LenthPSE(x)

説明

ベクトル x の値から Lenth の擬似標準誤差を求める。

引数

x ベクトル

Max()

「Maximum(var1, var2, ...)」（229ページ）を参照してください。

Maximum(var1, var2, ...)

Max(var1, var2, ...)

説明

引数の中での最大値、または、1 つの行列またはリスト内の最大値を戻す。複数の引数を指定する場合は、すべてを数値またはすべてを文字列にする必要があります。

Mean(var1, var2, ...)

説明

指定された複数の変数における、平均を戻す。

Min

「Minimum(var1, var2, ...)」（229ページ）を参照してください。

Minimum(var1, var2, ...)

Min(var1, var2, ...)

説明

引数の中での最小値、または、1 つの行列またはリスト内の最小値を戻す。複数の引数を指定する場合は、すべてを数値またはすべてを文字列にする必要があります。

N Missing(expression)

説明

指定された複数の変数における、欠測値の個数を戻す。

Number(var1, var2, ...)

説明

指定された複数の変数における、非欠測値の個数を戻す。

Product(i=initialValue, limitValue, bodyExpr)

説明

limitValue になるまで、すべての *i* について *bodyExpr* の結果を乗算し、積を戻す。

Quantile(p, arguments)

説明

引数の分位点 *p* を戻す。最初の引数には、0 ～ 1 のスカラー値または行列を指定できます。arguments の引数も、1 つの行列または 1 つのリストとして指定できます。

Std Dev(var1, var2, ...)

説明

指定された複数の変数における、標準偏差を戻す。

Sum(var1, var2, ...)

説明

指定された複数の変数における、合計を戻す。「Sum(.,.)」のように、すべての引数が欠測値の場合は、欠測値を戻します。

SSQ(x1, ...)

説明

全要素の平方和を戻す。引数には数値、行列、リストを指定できます。スカラー値が戻されます。欠測値は除外されます。

Summarize(<dt>, <by>, <count>, <sum>, <mean>, <min>, <max>, <stddev>, <corr>, <quantile>, <first>)

説明

データテーブルの要約統計量を求め、グローバル変数に格納する。

戻り値

なし

引数

dt（オプション）位置指定引数。データテーブルへの参照。この引数が割り当ての形式をとらない場合は、データテーブルの式とみなされます。

その他の引数はすべてオプションで、任意の順序で指定できます。通常、各引数は変数に割り当てられるので、値の表示や、さらなる操作が可能です。

name=By(col | list | Eval) By を指定すると、全体に対する 1 つの結果ではなく、By に指定した列の各グループごとに結果が計算される。

Summarize YByX(X(<x columns>, Y (<y columns>), Group(<grouping columns>), Freq(<freq column>), Weight(<weight column>))

説明

大規模なデータセットに対し、すべての組み合わせで二変量の関係の統計量を計算する。

戻り値

Y と X の組み合わせごとの *p* 値と対数価値のデータテーブル。『予測モデルおよび発展的なモデル』の「応答スクリーニング」章を参照してください。

引数

X(col) あてはめるモデルで使用する因子列。
Y(col) あてはめるモデルで使用する応答列。
Group(gcol) あてはめるモデルで使用するグループ化列。
Freq(col) あてはめるモデルで使用する（各行の）度数列。
Weight(col) あてはめるモデルで使用する重要度（影響度）の列。

ノート

「応答のスクリーニング」プラットフォームと同じ働きをします。詳細については、『予測モデルおよび発展的なモデル』の「応答スクリーニング」章を参照してください。

Summation(init, limitvalue, body)

説明

init から *limitvalue* までのすべての整数について、指定された式（body）の結果を合計し、その値を戻す。

Tolerance Limit(1-alpha, p, n)

説明

標本サイズ *n* の標本から計算される平均のうち割合 *p* だけが含まれるような区間を、信頼水準 (*1-alpha*) で求める。

超越関数

Arrhenius(n)

説明

温度 *n* を、Arrhenius モデルにおける説明変数の値に変換する。

戻り値

11605/(n+273.15)

引数

n 温度（単位は摂氏）

ノート

これは頻繁に使用される変換の1つです。

Arrhenius Inv(n)

説明

Arrenhius 関数の逆関数。値 *n* を摂氏の温度に変換する。

戻り値

11605/n-273.15

引数

n Arrhenius モデルにおいて説明変数の値に変換された値。

ノート

これは頻繁に使用される変換の1つです。

Beta(a, b)

説明

$$\frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

戻り値

ベータ関数

引数

a, b 数値

Digamma(n)

説明

ガンマ関数（LGamma）の対数の導関数。

戻り値

n におけるディガンマ関数の値。

引数

n 数値

Exp(a)

説明

e を *a* 乗する。

戻り値

e^a

引数
a 数値

等価表現
e()^a

ExpM1(x)

説明
Exp(x)-1 を戻す。*x* が非常に小さい場合に、より正確な計算結果を戻します。

Factorial(n)

説明
1 から *n* までの全整数を掛ける。

戻り値
n の階乗

引数
n 任意の整数

ノート
指定できる引数は 1 つだけです。

FFT(list, <named arguments>)

説明
行列のリストに対して高速 Fourier 変換（FFT）を行う。

戻り値
複素数を表す 1 つまたは複数の行列のリストを引数とし、その最初の引数と同じ次元の、2 つの行列のリストを戻す。

引数
list 1 つまたは 2 つの行列を含むリスト。1 つの行列で構成されている場合、その行列は実数とみなされます。2 つの行列で構成されている場合、1 番目は実数、2 番目は虚数部分とみなされます。2 つの行列は次元が同じで、どちらも行が 2 つ以上なければなりません。

名前付き引数
名前付き引数はすべてオプションです。
<<inverse(Boolean) 1（真）の場合、逆 FFT が実行される。
<<multivariate(Boolean) 1（真）の場合、多変量 FFT が実行される。0（偽）の場合、空間 FFT が実行される。
<<scale(number) 戻り値に *number* で指定した乗数を掛け合わせる。

Fit Transform To Normal(Distribution("name"), Y(vector), <Freq(vector))

説明

データのベクトルに対し、正規分布へと変換するための分布をあてはめる。Johnson Sl、Johnson Sb、Johnson Su、一般化対数（GLog）といった分布をあてはめることができます。

戻り値

パラメータ推定値、共分散行列、対数尤度、AICc、収束メッセージ、変換値で構成されるリスト

Gamma(t, <limit>)

説明

x に対するガンマ関数の値を戻す。

$$\Gamma(t) = \int_0^{\infty} x^{t-1} e^{-x} dx$$

戻り値

ガンマ関数の値

ノート

Gamma(t, limit) は Gamma(t) と積分される式は同じですが、積分の範囲における上限を無限ではなく limit とします。

引数

t 数値または列
limit（オプション）上限。デフォルトは∞

LGamma(t)

説明

t の対数ガンマ関数（ガンマ関数の自然対数）を戻す。

Ln(n)

LnZ(n)

説明

n の自然対数（底 *e* の対数）を戻す。

ノート

LnZ() は、数値微分を伴う分析で内部的に使用されます。Ln(n) 関数を代わりに使用することを推奨します。

Log(n, <base>)

説明

n の自然対数（底 *e* の対数）を戻す。オプションの第 2 引数を追加すると、別の底を指定できます。たとえば、Log(n,3) は、3 を底とする *n* の対数です。Log の引数は、任意の数式をとることができます。式 Log(e()) は 1、Log(32,2) は 5 です。

Log10(n)

説明

n の常用対数（底は 10）を戻す。

Log1P(n)

説明

x がきわめて小さいときに精度がより高くなることを除けば、Log(1 + *x*) と同じ。

Logit(p)

説明

log(p/(1-p)) を戻す。

N Choose K(n, k)

説明

この関数は、一度に *n* 個のうちから *k* 個を選択する場合の組み合わせの数（「*n*C*k*」）を戻す。この組み合わせの数は、$n!/(k!(n-k)!)$ のように階乗を使った式で計算することができます。たとえば、NChooseK(5,2) は 10 になります。

ノート

この関数の計算において、JMP では LGamma 関数が内部的に使用されています。そのため、結果は必ずしも整数とは限りません。

Power(a, <b>)

a^b

説明

a を *b* 乗する。

戻り値

a を *b* 回掛け合わせた積

引数

a　変数、数値、または行列。
b　（オプション）変数または数値。

ノート

Power() の場合、第 2 引数（*b*）はオプションで、デフォルト値は 2 です。Power(a) は a^2 を戻します。

Root(n, <r>)

説明

n の *r* 次の根を戻す。デフォルトでは *r* が 2 で、平方根を戻します。

SbInv(z, gamma, delta, theta, sigma)

説明

Johnson Sb 逆変換。引数が正規分布の場合、戻り値は Johnson Sb 分布です。

SbTrans(x, gamma, delta, theta, sigma)

説明

Johnson Sb 変換。上下に有界な Johnson Sb 分布の変数を、平均 0、分散 1 の正規分布に変換します。

Scheffe Cubic(x1, x2)

説明

x1*x2*(x1-x2) の結果を戻す。3 次の配合モデルの表記に対応しています。

SlInv(z, gamma, delta, theta, sigma)

説明

Johnson Sl 逆変換。引数が正規分布の場合、戻り値は Johnson Sl 分布です。

SlTrans(x, gamma, delta, theta, sigma)

説明

Johnson Sl 変換。上もしくは下に有界な Johnson Sl 分布の変数を、平均 0、分散 1 の正規分布に変換します。

Sqrt(n)

説明

n の平方根を戻す。

Squash(expr)

説明

関数 1/ [1 + exp(*expr*)] を効率よく計算する。

Squish(expr)

説明

Squash(-expr) または $1/(1+e^{-\text{expr}})$ と同じ。

SuInv(z, gamma, delta, theta, sigma)

説明

Johnson Su 逆変換。引数が正規分布の場合、戻り値は Johnson Su 分布です。

SuTrans(x, gamma, delta, theta, sigma)

説明

Johnson Su 変換。Johnson Su 分布の変数を、平均 0、分散 1 の正規分布に変換します。

Trigamma()

説明

n におけるトリガンマ関数の値を戻す。トリガンマ関数はディガンマ関数の導関数です。

三角関数

JMP の三角関数では、すべての角度引数をラジアンで入力します。

ArcCosH(x)

説明

逆双曲余弦関数の値を戻す。

戻り値

x に対する逆双曲余弦関数の値

引数

x 任意の数値、数値変数、または数値式。

ArcCosine(x)

ArCos(x)

説明

逆余弦関数の値を戻す。

戻り値

x に対する逆余弦関数の値。戻り値は、角度でラジアン単位

引数

x 任意の数値、数値変数、または数値式。

ArcSine(x)

ArSin(x)

説明

逆正弦関数の値を戻す。

戻り値

x に対する逆正弦関数の値。戻り値は、角度でラジアン単位

引数

x 任意の数値、数値変数、または数値式。

ArcSinH(x)

説明

逆双曲正弦関数の値を戻す。

戻り値

x に対する逆双曲正弦関数の値

引数

x 任意の数値、数値変数、または数値式。

ArcTangent(x1, <x2=1>)

ArcTan(x1 <x2=1>)

ATan(x1 <x2=1>)

説明

逆正接関数の値を戻す。

戻り値

正接関数（*x1/x2*）の逆関数を戻す（範囲は -Pi()/2 ～ Pi()/2）。

引数

x1 任意の数値、数値変数、または数値式。

x2=1 *atan2* を指定する。

ArcTanH(x)

説明

逆双曲正接関数の値を戻す。

戻り値

x に対する逆双曲正接関数の値

引数

x 任意の数値、数値変数、または数値式。

CosH(x)

説明

双曲余弦

戻り値

x の双曲余弦

引数

x 任意の数値、数値変数、または数値式。

Cosine(x)

Cos(x)

説明

余弦

戻り値

x の余弦

引数

x 任意の数値、数値変数、または数値式。ラジアン単位での角度。

Sine(expr)

Sin(expr)

説明

正弦

SinH(expr)

説明

双曲正弦

Tangent(expr)

Tan(expr)

説明

正接

TanH(expr)

説明

双曲線正接

ユーティリティ関数

Add(a, b, ...)

a+b+...

説明

リストされた引数の値を加算する。どの引数も変更されません。

戻り値

列の合計

引数

Add(): カンマで区切った、複数の変数、数値、または行列。

a+b: 任意の変数、数値、または行列。

ノート

引数はいくつでも使えます。引数が指定されていない場合、Add() は 0 を戻します。

次も参照

『スクリプトガイド』の「データ構造」章

Batch Interactive(Boolean)

説明

JSL の実行時環境を、バッチモードまたは対話型モードに設定する。

戻り値

前の文字列

例

```
Batch Interactive(1);      // バッチモード 1 に設定
   <code>
Batch Interactive(0);      // 前のモードに戻す
```

ノート

この関数によって実行環境をバッチモードにすると、メッセージが対話型のダイアログではなく、ログに出力されるようになります。この機能は、ジャーナルでスクリプトを使用するときに役立ちます。

Beep()

説明

警告音を鳴らす。

戻り値

なし

Blob MD5(blob)

説明

引数 *blob* から、16 バイトの BLOB を生成する。

ノート

戻り値の 16 バイトの BLOB は、引数の BLOB から計算された MD5 チェックサム（ハッシュ値）です。

Blob Peek(blob, offset, length)

説明

引数 *blob* から指定されたバイトだけ抜き出し、新しい BLOB を作成する。

戻り値

BLOB オブジェクト

引数

`blob` BLOB（binary large object）

`offset` BLOB の先頭から何バイト目までを抜き出すかを指定する整数。最初のバイトはオフセットが 0 で、2 番目のバイトはオフセットが 1

`Length` オフセットから何バイト目までを抜き出すかを指定する整数

Build Information()

説明

ビルト日時、リリースビルトとデバッグビルトの区別、および製品名を、カンマで区切った文字列で戻す。

Caption({*h*, *v*}, "text", <Delayed(seconds)>, <Font(font)>, <Font Size(size)>, <Text Color("color")>, <Back Color("color")>, <Spoken(Boolean)

説明

指定のテキスト（*text*）を含んだキャプションウィンドウを指定の位置 *{h,v}* に表示する。キャプションの表示を *seconds* だけ遅らせたり、音声を出したりすることもできます。フォントの種類、サイズ、色、および背景色を指定することもできます。

戻り値

なし

引数

`{h, v}` 2 つの値のリスト。*h* は、モニタの左上からの横方向の位置をピクセルで表したもの。*v* は、モニタの左上からの縦方向の位置をピクセルで表したもの

`text` キャプションウィンドウに表示される引用符付き文字列または文字列への参照

`Delayed(seconds)`（オプション）*seconds* は、文字列をキャプションウィンドウに表示するまでの遅延時間。このオプションを設定すると、このキャプションおよび後続のすべてのキャプションは指定の秒数が経ってから表示されます。

`Font(font)` フォントの種類を指定する。

`Font Size(size)` フォントのサイズを指定する。

Text Color("color") テキストの色を指定する。
Back Color("color") 背景色を指定する。
Spoken(Boolean) テキスト（*text*）を音声で読み上げる。現在の設定（オンまたはオフ）は、Spoken 設定を含む別の Caption ステートメントによって変更されるまで有効となります。

Datafeed()

「Open Datafeed()」（250ページ）を参照してください。

Debug Break()

JSL デバッガが開いているとき、この関数はスクリプト内のその時点で JSL スクリプトの実行を停止します。この関数は、ユーザが指定した条件の下で、デバッガで追跡調査をしているときに便利です。JSL デバッガが起動していない場合、この関数は実行されません。

Decode64 Double("string")

説明
Base64 でエンコードされた文字列から、浮動小数点数を戻す。

戻り値
浮動小数点数

引数
string Base64 でエンコードされた文字列

Divide(a, b)

Divide(x)

a/b

説明
a を *b* で割る。引数が 1 つだけの場合（divide(x)）、1 を *x* で割ります。

戻り値
a/*b* の商、または、引数が 1 つだけの場合は、*x* の逆数 (1/*x*)

引数
a, b, x 変数、数値、または行列。

ノート
両引数が行列のときは、行列の割り算を行います。

次も参照
『スクリプトガイド』の「データ構造」章

Empty()

説明
何もしない。計算式エディタで空のボックスを作成するのに使用されます。

戻り値
空の値

引数
なし

Encode64 Double(n)

説明
浮動小数点数から Base64 でエンコードされた文字列を戻す。

戻り値
Base64 でエンコードされた文字列

引数
n 浮動小数点数

Faure Quasi Random Sequence(nDim, nRow)

説明
Faure 数列を使って、Space Filling の擬似乱数シーケンスを生成する。

Get Addin("id")

説明
id によって指定された登録済みアドインを取得する。

戻り値
アドイン用スクリプト可能オブジェクト。指定された ID のアドインが見つからない場合は空を戻します。

引数
"id" インストールされたアドインの ID

Get Addins()

戻り値
登録されているアドインすべてのリスト

Get Addr Info("address", <port>)

説明
指定されたアドレス名を、数値アドレスに変換する

戻り値

文字列のリスト。最初の要素はコマンド名（Get Addr Info）。2 番目は結果（たとえば、コマンドが成功した場合は "ok"）。3 番目はいくつかの情報を表す文字列のリスト。この 3 番目のリストのなかに、指定した名前に対応する数値アドレスが含まれている。

引数

address 名前を指定する引用符付き文字列（たとえば、"www.sas.com"）。
port アドレスのポート

Get Clipboard()

説明

コンピュータのクリップボードからテキストを戻す。クリップボードの内容がテキストではない場合、結果はヌルです。

Get Name Info("address", <port>)

説明

指定された数値アドレスを、アドレス名に変換する

戻り値

文字列のリスト。最初の要素はコマンド名（GetNameInfo）。2 番目は結果（たとえば、コマンドが成功した場合は "ok"）。3 番目はいくつかの情報を表す文字列のリスト。この 3 番目のリストのなかに、指定した数値アドレスに対応する名前が含まれている。

引数

address 数値アドレスを指定する引用符付き文字列（たとえば、"149.173.5.120"）。
port アドレスのポート

Get Platform Preferences(<platform <(option, ...)> ... >)

Get Platform Preference(<platform <(option, ...)> ... >)

説明

指定されたプラットフォームの環境設定を戻す。

戻り値

プラットフォームの環境設定のリスト

引数

platform（オプション）プラットフォーム名。これが指定されていない場合、指定したプラットフォームのすべてのオプションの値を戻します。プラットフォームにつき、1 つまたは複数の環境設定を指定できます。

option（オプション）オプションの値。これが指定されていない場合、すべてのプラットフォーム環境設定の値を戻します。

ノート

表 2.3 に、プラットフォーム環境設定を取得するための構文を示します。

表2.3 Get Platform Preferences()の構文

構文	説明
Get Platform Preferences( )	すべてのプラットフォーム環境設定の現在のオプションの値を戻す。
Get Platform Preferences(Platform)	指定されたプラットフォーム環境設定の現在のオプションの値を戻す。
Get Platform Preferences(Platform(Option))	指定されたプラットフォームの現在のオプションの値を戻す。
Get Platform Preferences(<<Changed)	すべてのプラットフォーム環境設定について、変更されたオプションの現在の値を戻す。
Get Platform Preferences(Platform(<<Changed))	指定されたプラットフォーム環境設定について、変更されたオプションの現在の値を戻す。
Get Platform Preferences(Platform(Option (<<Changed)))	指定されたプラットフォーム環境設定について、変更されたオプションの現在の値を戻す。

例

JMPのプラットフォームウィンドウまたはスクリプトを通じて、ユーザが複数の環境設定を変更したとします。

```
Platform Preferences(
   Distribution( Set Bin Width( 2 ), Horizontal Layout( 1 ) ),
   Model Dialog( Keep Dialog Open( 1 ) ),
   Graph Builder( Legend Position( "Bottom" ) )
);
```

変更されたすべてのプラットフォーム環境設定を取得するには、次のように Get Platform Preferences(<<Changed) を使用します。

```
Get Platform Preferences( <<Changed );
  Platform Preferences(
    Distribution( Horizontal Layout( 1 ), Set Bin Width( 2, <<On ) ),
    Graph Builder( Legend Position( "Bottom", <<On ) ),
    Model Dialog( Keep dialog open( 1 ) )
  );
```

Get Preferences(<preference_name>)

Get Preference(<preference_name>)

説明

指定された環境設定の設定を戻す。

戻り値

環境設定の設定のリスト

引数

preference_name（オプション）環境設定名。これが指定されていない場合、すべての環境設定を戻します。指定された場合は、その環境設定の設定を戻します。

ノート

テキストデータファイル、Windows のみ、Mac OS の設定、フォント、通信、スクリプトエディタ、JMP アップデートの環境設定には JSL を使ってアクセスすることはできません。プラットフォーム環境設定の取得の詳細については、「Get Platform Preferences(<platform <(option, ...)> ...>)」（244 ページ）を参照してください。

Glue(expr1, expr2, ...)

expr1; expr2

説明

各引数を順番に評価する。

戻り値

最後に評価された引数の結果

引数

1 つ以上の有効な JSL 式

ノート

Glue() では、式の区切りとしてセミコロンの方が一般的に使用されます。

Host Is(argument)

説明

ホスト環境が、指定の OS と一致しているかどうかを判断する。

戻り値

現在の OS 環境が引数と一致するときは 1（真）、そうでなければ 0（偽）

引数

引数 Windows または Mac のいずれかのオペレーティングシステム。コンピュータが 32 ビットか 64 ビットかを判断するには、Bits32 または Bits64 を指定します。

ノート

一度に 1 つの引数しかテストできません。無効な引数を指定した場合は、0（偽）が戻されます。

Is Alt Key()

説明

Alt キーが押されているときは 1、そうでなければ 0 を戻す。

ノート

Macintosh の場合、Is Alt Key() は option キーをテストします。

Is Command Key()

説明

コマンドキーが押されているときは1、そうでなければ0を戻す。

Is Context Key()

説明

コンテキストキーが押されているときは1、そうでなければ0を戻す。

Is Control Key()

説明

Ctrlキーが押されているときは1、そうでなければ0を戻す。

ノート

Macintoshの場合、`Is Control Key()`はcommandキーをチェックします。

Is Option Key()

説明

Optionキーが押されているときは1、そうでなければ0を戻す。

Is Shift Key()

説明

Shiftキーが押されているときは1、そうでなければ0を戻す。

JMP Product Name()

説明

JMPの製品ライセンスに応じて`"Standard"`、`"Pro"`、または`"Student"`を戻す。

JMP Version()

説明

使用しているJMPのバージョンを戻す。

戻り値

release.revision<.fix>の文字列

引数

なし

Load DLL("path" <,AutoDeclare(Boolean | Quiet | Verbose)>)

Load DLL("path" <, Quiet | Verbose>)

説明

パス（*path*）で指定されたDLLをロードする。

引数

path DLLをロードする場所のパス名。

AutoDeclare(Boolean | Quiet | Verbose)（オプション）AutoDeclare(1)およびAutoDeclare(Verbose)は、ログにverboseメッセージを書き込む。AutoDeclare(Quiet)は、ログウィンドウのメッセージを無効にします。このオプションを省略すると、ログにverboseメッセージが書き込まれます。

Quiet | Verbose（オプション）Declare Function()を使用した場合、このオプションは、ログウィンドウへのメッセージ出力のオフ（Quiet）とオン（Verbose）を切り替える。

次も参照

DLLがロードされたら、DLLオブジェクトメッセージを送り、それを操作します。メッセージの詳細については、「JSLメッセージ」章の「ダイナミックリンクライブラリ（DLL）」（320ページ）を参照してください。例については、『スクリプトガイド』の「JMPの拡張」章を参照してください。

Mail("address", "subject", "message", <"attachment filepath" | {"attachment 1 filepath", "attachment 2 filepath", ...}>)

説明

（Windows）*address*で指定されたメールアドレスに、指定のタイトル（*subject*）とメッセージ（*message*）から成る電子メールを送る（MAPIを使用）。オプションで*attachment*引数を指定すると、1つまたは複数の添付ファイルを送信できます。attachment引数には、文字列または文字列のリストを指定できます。

（Macintosh）ユーザのメールアプリケーション内で電子メールを作成する。ユーザは、メールのウィンドウで［**送信**］をクリックしなければなりません。Mountain Lionの場合は、オペレーティングシステムの制限のため、メールのウィンドウでアドレスと件名を入力しなければなりません。Macintosh版のMicrosoft Outlookの場合は、メールに手動で添付ファイルを追加する必要があります。

例

複数のファイルを添付して、メールを送信するには、次のように指定します。

```
Mail(
   "yourname@company.com",
   "最新データとスクリプト",
   "本日更新のデータテーブルとスクリプトを添付します。",
   {"$DOCUMENTS/wd.jsl", "$DOCUMENTS/survey.jmp"}
);
```

もしくは、次のように指定します。

```
list = {"$DOCUMENTS/wd.jsl", "$DOCUMENTS/survey.jmp"};
Mail(
   "yourname@company.com",
```

```
        " 最新データとスクリプト ",
        " 本日更新のデータテーブルとスクリプトを添付します。",
        リスト
    );
```

ノート

Macintosh の場合、Mail() は Mountain Lion およびそれ以降のオペレーティングシステムで機能します。

Main Menu(string, <string>)

説明

引用符付き文字列（*string*）で示された JMP メニューのコマンドを実行する。

引数

string メニューエディタに表示される、項目の内部名。たとえば、［ファイル］メニューの［新規作成］コマンドに対応する内部名は、“NEW” です。

string（オプション）コマンドの送り先であるウィンドウの名前。

例

Main Menu() には、完全パスまた部分パスを指定できます。部分パス名に一致するメニュー項目が複数ある場合は、最初に見つかったメニュー項目が実行されます。検索は、上位メニュー（ファイル、テーブル、DOE など）から下位メニューの順に行われます。

```
Main Menu( "File:New:Data Table" ); // 完全パス
Main Menu( "Data Table" ); // 部分パス
```

Minus(a)

-a

説明

a の符号を反転する。

戻り値

a が正のときは -Abs(*a*)（a=3; -a=-3; Minus(a)=-3）

a が負のときは Abs(*a*)（a=-3; -a=3; Minus(a)=3）

a が 0 のときは 0（a=0; -a=0; Minus(a)=0）

a が割り当てられていないときは欠測値

引数

a 変数または数値。変数には数値または行列が含まれている必要がある。

Multiply(a, b, ...)

a*b*...

説明

すべての値を掛ける。どの引数も変更されません。

戻り値
n の階乗

引数
任意の変数、数値、または行列。

ノート
引数はいくつでも使えます。引数が指定されていない場合、Multiply() は 1 を戻します。

次も参照
『スクリプトガイド』の「データ構造」章

Open Datafeed()

Datafeed()

説明
データフィードオブジェクトおよびウィンドウを作成する。

戻り値
データフィードオブジェクトへの参照

引数
引数は必要ありません。ただし、通常は、Open Datafeed() 内の引数によって、データフィードの基本操作を設定します。

Parse XML(string, On Element("tagname", Start Tag(expr), End Tag(expr)))

説明
On Element で指定された XML タグによって、XML を解析する。

Platform Preferences(platform(option(value)), ...)

Platform Preference(platform(option(value)), ...)

Set Platform Preferences(platform(option(value)), ...)

Set Platform Preference(platform(option(value)), ...)

説明
プラットフォームの環境設定を設定または解除する。

引数
platform 環境設定を行うプラットフォーム。
option オプションの名前。
value オプションの値。

ノート
表 2.4 に、プラットフォームの環境設定を行うための構文を示します。

表2.4 Platform Preferences()の構文

構文	説明
Platform Preferences(<<Default) Platform Preferences(<<Factory Default) Platform Preferences(Default)	すべてのプラットフォーム環境設定をデフォルトの値にリセットする。
Platform Preferences(Platform(<<Default)) Platform Preferences(Platform(<<Factory Default)) Platform Preferences(Platform(Default))	指定されたプラットフォーム環境設定をデフォルトの値にリセットする。
Platform Preferences(Platform(option (<<Default))) Platform Preferences(Platform(option (<<Factory Default)))	指定されたプラットフォームオプションをデフォルトの値にリセットする。
Platform Preferences(Platform(option(value)))	指定されたプラットフォームオプションの値を設定する。デフォルトはオンです。
Platform Preferences(Platform(option))	指定されたプラットフォームオプションをデフォルトの値にリセットする。
Platform Preferences(Platform(option(<<On)))	指定されたプラットフォームオプションをオンにする。
Platform Preferences(Platform(option(<<Off)))	指定されたプラットフォームオプションをオフにする。
Platform Preferences(Platform(option(value, <<On)))	指定されたプラットフォームオプションをオンにし、値を設定する。
Platform Preferences(Platform(option(value, <<Off)))	指定されたプラットフォームオプションをオフにし、値を設定する。

例

次の式は、「一変量の分布」環境設定の［棒の幅の設定］オプションを選択（オンに）して、値を2に設定します。

```
Platform Preferences( Distribution( Set Bin Width( 2 ) ) );
```

次の式は、［棒の幅の設定］の値を変更し、このオプションを無効にします。

```
Platform Preferences( Distribution( Set Bin Width( 2, <<Off ) ) );
```

次の式は、［棒の幅の設定］の値をデフォルト値にリセットして、環境設定の選択を解除します。

```
Platform Preferences( Distribution( Set Bin Width( <<Default ) ) );
```

Polytope Uniform Random(samples, A, b, L, U, neq, nle, nge, <nwarm=200>, <nstride=25>)

説明

凸多面体上に一様乱数の点を生成する。

引数

Samples 生成される乱数の個数
A 制約式の係数の行列
B 制約式の右辺値
L, U 変数の下限および上限
neq 等号制約式の数
nle「以下」を示す不等号制約式の数
nge「以上」を示す不等号制約式の数
nwarm（オプション）予備反復の回数。結果を出力する前に、何回、反復を行うかを指定します。
nstride（オプション）結果を出力する間隔。何回の 1 回のペースで結果を出力するかを指定します。

ノート

制約式は、等号制約、「以下」を示す不等号制約、「以上」を示す不等号制約の順に指定しなければなりません。

Preferences(pref1(value1), ...)

Preference(pref1(value1), ...)

Pref(pref1(value1), ...)

Prefs(pref1(value1), ...)

Set Preferences(pref1(value1), ...)

Set Preference(pref1(value1), ...)

説明

JMP の環境設定を設定する。

引数

Add Files Opened by Scripts to the Recent Files List(Boolean) スクリプトが開いたファイルをホームウィンドウの「最近使ったファイル」リストに追加するかどうかを指定する。
Analysis Destination(window) 新しい分析の結果をどこに出力するかを指定する。
Antation Font("font", size, "style") レポートに表示される注釈のフォントを指定する。
Axis Font("font", size, "style") 軸ラベルのフォントを指定する。
Axis Title Font("font", size, "style") 軸タイトルのフォントを指定する。
Background Color({R, G, B} | <color>) ウィンドウの背景色を指定する。
Calculator Boxing(Boolean) 計算式エディタの中にボックスを作成し、大きい式の中の各項の階層がわかるようにする。
Data Table Font("font", size, "style") データテーブルのフォントを指定する。
Data Table Title on Output(Boolean) レポートにデータテーブル名のタイトルをつける。
Date Title on Output(Boolean) レポートに現在の日付のタイトルをつける。

Evaluate OnOpen Scripts("always"|"never"|"prompt") ユーザがデータテーブルを開いたときに On Open テーブルスクリプトを実行するかどうかを指定する。デフォルトでは、確認のメッセージが表示されます。1 つの JMP セッションで再度同じデータテーブルを開いた場合は、前回のユーザーの選択が適用されます。ただし、別のプログラムを実行するスクリプトは実行されません。

Excel Has Labels(Boolean) オンにすると、データの先頭行が列見出しと解釈される。

Excel Selection(Boolean) オンにすると、Excel ワークブックからどの（空白ではない）Excel ワークシートを読み込むか指定するよう、プロンプトが表示される。

File Location Settings(<Directory Type>("<path>"<,"initial directory">)) 次のディレクトリを指定できます。

Data Files Directory: データファイルのデフォルトの保存場所を設定する。

Help Files Directory: ヘルプファイルの保存場所を設定する。

Installation Directory: Windows では、次の場所が JMP のデフォルトのインストールフォルダに設定されている。

"C:/Program Files/SAS/JMP/13"、"C:/Program Files/SAS/JMPPro/13"、または "C:/Program Files/SAS/JMPSW/13"

License File Path: JMP ライセンスファイルのデフォルトの保存場所を設定する。

Preferences File Directory: 環境設定ファイルの保存場所を設定する。

Save As Directory: 名前を付けて保存の操作で使われるデフォルトの保存場所を設定する。

Foreground Color(color) ウィンドウの前面の色を指定する。

Formula Font("font", size, "style") 計算式エディタに用いるフォントを指定する。

Graph Background Color(color) グラフのフレーム内の背景色を指定する。

Graph Marker Size(size) マーカーを描くときのデフォルトの大きさを指定する。

Heading Font("font", size, "style") レポート中の表の、列見出しに使われるフォントを指定する。

Initial JMP Starter Window(Boolean) 起動時の「JMP スターター」ウィンドウの表示／非表示を指定する。

Initial Splash Window(Boolean) 起動時スプラッシュ画面の表示／非表示を指定する。

Maximum JMP Call Depth(size) JMP 呼び出しの最大の深さ（スタックサイズ）のデフォルト値を設定する。JMP が作成する各スレッドは、デフォルトで 2MB のスタックがあります。呼び出しの最大の深さをリセットすると、スタックが物理的にオーバーフローすることがあります。

Marker Font("font", size, "style") プロットで使用されるマーカーのフォントを指定する。

Monospaced Font("font", size, "style") モノスペースフォントを指定する。

ODBC Suppress Internal Quoting(Boolean) SQL ステートメントに含まれるテーブル名や変数名に大文字、小文字、スペースが混在している場合に、内部で引用符を追加しない。

Outline Connecting Lines(Boolean) 同じレベルのアウトラインノードのタイトルを線でつないで見やすくする。

Print Settings(option(value), ...)「ページ設定」ウィンドウの印刷オプションを変更する。

Margins(<n>, <n>, <n>, <n>): 左、上、右、下の余白を設定する。センチ単位で指定します。

Margins(<n>): すべての余白を同じ値（単位はセンチ）に設定する。

Orientation("portrait" | "landscape"): 用紙の向きを変更する。

Headers(<"char">, <"char">, <"char">): 左、中央、右のヘッダに表示されるテキストを指定する。

Headers(<"char">): ヘッダに表示するテキストのみを指定する。

Footers(<"char">, <"char">, <"char">): 左、中央、右のフッタに表示されるテキストを指定する。

Footers(<"char">): フッタに表示するテキストのみを指定する。

Scale(<n>): 印刷倍率を増減する。

Show Explanations(Boolean) 分析の種類によっては、出力について説明するテキストをオプションで表示する。

Show Menu Tips(Boolean) メニューのヒントの表示／非表示を指定する。(Windows のみ)。

Show Status Bar(Boolean) ステータスバーの表示／非表示を指定する。

Small Font("font", size, "style") 小さなテキストに用いるフォントを指定する。

Text Font("font", size, "style") レポートの中の一般的なテキストのフォントを指定する。

Thin Postscript Lines(Boolean) (Macintosh のみ) ポストスクリプトプリンタに出力する場合の線の太さを、それ以外の場合より細くするよう指定する。

Title Font("font", size, "style") タイトルのフォントを指定する。引数にはフォント名 (例 : "Times")、ポイント数、スタイル ("bold"、"plain"、"underline"、"italic") があります。

Use Triple-S Labels as Headings(Boolean) オンにすると、ラベル名が列見出しと解釈される。

例

次の式は、すべての環境設定をデフォルトの値にリセットします。

```
Preferences( "Default" );
Preferences( "Factory Default" );
```

ノート

テキストデータファイル、Windows のみ、Mac OS の設定、フォント、通信、スクリプトエディタ、JMP アップデートの環境設定には JSL を使ってアクセスすることはできません。プラットフォーム環境設定の設定の詳細については、「Platform Preferences(platform(option(value)), ...)」(250 ページ) を参照してください。

Register Addin("unique_id", "home_folder", <named_arguments>)

説明

JMP アドインを登録し、正常に登録されたら、そのアドインをロードする。

戻り値

成功した場合は、登録されたアドインを示すスクリプト可能オブジェクトを戻す。成功しなかった場合は、空を戻す。

引数

unique_id アドインの一意の識別子を示す引用符付き文字列。最大 64 文字まで使用できます。文字列の最初は文字でなければならず、文字、数字、ピリオド、下線を使用できます。一意である確率を上げるために、DNS の逆の名前を使用することをお勧めします。

home_folder アドインファイルを含むフォルダのファイルパスを示す引用符付き文字列。ファイルパスは、ホストのオペレーティングシステムのファイルパス要件を満たすものでなければなりません。

DisplayName("name")（オプション）制限のある一意の ID 名ではなく、JMP ユーザインターフェースでのアドインの表示に使われる、わかりやすい表示名。

JMPVersion("version")（オプション）JMP の特定のバージョンを示す引用符付き文字列。デフォルト値は "All" で、この場合、アドインに対応しているすべての JMP バージョンでアドインのロードおよび実行ができます。"Current" を指定すると、アドインの使用が現在のバージョンのみに制限されます。引用符付きのバージョン番号（"7" や "9" など）を指定すると、アドインの使用が単一の JMP バージョンに制限されます。

LoadsAtStartup(Boolean)（オプション）ブール値。デフォルト値は 1（真）で、この場合、JMP の起動時にアドインがロードされます。0（偽）を指定すると、アドインは自動的にはロードされません。

LoadNow(Boolean) アドインを即座にロードする。

ノート

指定されたホームフォルダに「addin.def」という名前のファイルがあれば、Register Addin() 関数に含まれていないオプションの引数すべてに、そのファイルの値が使用されます。

例

次の例では、第 1 引数で一意の識別子を指定し、第 2 引数でアドインのインストール先を指定しています。第 3 引数では、アドイン名が通常表示される場所で使用する表示名を指定しています（たとえば、Windows の［**表示**］>［**アドイン**］）。

```
Register Addin("com.company.lee.dan.MyAddIn","$DOCUMENTS/myaddin", displayname(
   "Calculator Addin" ));
```

第 2 引数は、$ADDIN_HOME パス変数を設定します。このアドインスクリプトを参照する際は、パス変数の末尾に必ずスラッシュを含めます。

```
Include("$ADDIN_HOME(com.jmp.jperk.texttocols)/texttocols.jsl");
```

Revert Menu()

説明

JMP メニューを出荷時の状態に戻す。

Run Program(Executable("path/filename.exe"), Options({"/a", "/b", "..."}), Read Function(expression), Write Function(expression), Parameter(expression))

説明

Executable 引数で指定された外部プログラムを、Options 引数で指定されたコマンドライン引数を使って実行する。

戻り値

文字列、BLOB、または Run Program オブジェクトのいずれかを、Read Function 引数による制御に従って戻す。

引数

Executable 実行ファイルのパス。Macintosh の場合は、実行ファイルのフルパスをタイプします。

Options 実行ファイルのコマンドライン引数。

Read Function Read Function("text") が指定された場合はテキスト文字列を戻し、Read Function("blob") が指定された場合は BLOB を戻す。スクリプトは、外部プログラムがその stdout を閉じるまで待ちます。その後、Run Program は、外部プログラムが文字列または BLOB として stdout に書き込んだすべてのデータを戻します。

Read Function が指定されなかった場合は、Run Program オブジェクトを戻します。

Write Function（オプション）関数を値として受け入れる。"text" や "blob" は受け入れられません。

Parameter（オプション）Read Function で式を読み／書きするための引数。

メモ: Read Function が指定されていない場合に戻される Run Program オブジェクトは、外部プログラムの stdout からデータを読み取るための以下のメッセージを受け入れます。

- <<Read: 読み取り可能なデータを文字列として読み取る。読み取り可能なデータがない場合は、空の文字列を戻します。
- <<Can Read: 読み取り可能なデータがある場合は 1（真）を戻す。
- <<Is ReadEOF: 外部プログラムが完了し、すべてのデータを読み取ったら、1（真）を戻す。

  これらのメッセージを使って、外部プログラムによってデータが生成されたときに、それらのデータをポーリングし、処理できます。

- Run Program オブジェクトは、データを外部プログラムの stdin に書き込むための以下のメッセージを受け入れます。
  - <<Write("text"): 外部プログラムの stdin にデータを送る。
  - <<Can Write: 外部プログラムが即時にデータを受け入れる場合は 1（真）を戻す。そうでない場合、<<Write を呼び出すとスクリプトが止まってしまいます。
  - <<WriteEOF: データ送信が終了したことを示す、外部プログラムへの信号。
- 戻された Run Program オブジェクトにメッセージを送る代わりに、Read Function 引数をインライン関数として指定できます。RP は Run Program オブジェクトです。

```
RP = Run Program(
   Executable( ... ),
   Read Function(
      Function( {RP},
         <ここにコードを入力>
         RP << Read
      )
   )
);
```

Parameter(optParm) 引数は、Read Function におけるオプションの引数です。これが指定された場合、Read Function と Write Function に定義された関数は、第 2 引数（optParm の値）を受け取ることができます。

例

次のスクリプトは、Write Function引数の一例です。RPはRun Programオブジェクトです。この例では、<<Writeと<<WriteEOFのメッセージを受け取っています。

```
RP = Run Program(
   Executable( ... ),
   Write Function(
      Function( {RP},
         < ここにコードを入力 >
         RP << Write( "Program finished." )
      )
   )
);
```

次のスクリプトは、Parameter(optParm)引数の一例です。

```
RP = Run Program(
   Executable( ... ),
   Parameter( x ),
   Read Function( Function( {RP, optParm},... ) )
);
```

Read Function内で、optParmは*x*の値を含んでいます。Parameter引数を指定していない場合は、関数内のoptParm引数にアクセスしようとしないでください。

Schedule(n, script)

説明

*n*秒後に指定のスクリプト（*script*）を実行するというイベントをキューに入れる。

Set Clipboard(string)

説明

指定された引数（"*string*"）のテキスト部分の情報をクリップボードに入れる。

SetJVMOption(Version("<version number>"))

説明

JMPで使用するJava Runtime Environemnt（JRE）のバージョン（JMPとともにインストールされるバージョン以外）を指定できる。このスクリプトは、JMPがJREに接続する前に実行する必要があります。

引数

version（Windowsのみ）Windowsレジストリの JavaSoft/Java Runtime Environment キーに関して次の2つの要件がある。1つは、有効なjvm.dllを指す「RuntimeLib」と言う文字列をキーに含める必要があります。もう1つは、Java Runtime Environmentキーに、引用符付きのJVMバージョン番号の名前を持つキーを含める必要があります。

Set Platform Preference()

Set Platform Preferences()

「Platform Preferences(platform(option(value)), ...)」(250ページ)を参照してください。

Set Preference()

Set Preferences()

「Preferences(pref1(value1), ...)」(252ページ)を参照してください。

Set Toolbar Visibility("toolbar name" | default | all, window type | all, "true" | "false")

説明

Windows で、ウィンドウタイプに基づいて、またはすべてのウィンドウに対して、ツールバーを表示するか、非表示にする。

引数

`toolbar name | default | all` ツールバーの内部名(JMP の［**表示**］>［**ツールバー**］のリストを参照)、指定したウィンドウタイプのデフォルトのツールバー(default)、またはすべてのツールバー(all)。`"toolbar name"` には引用符が必要です。

`window type | all` データテーブル(Data table)、スクリプト(script)、レポート(report)、ジャーナル(journal)、またはすべてのウィンドウ(all)。

`true | false` ツールバーの表示または非表示を指定する引用符付き文字列。

Shortest Edit Script()

説明

2つのリスト、文字列、行列、またはシーケンスを比較する。詳細な構文オプションについては、［**ヘルプ**］メニューの［スクリプトの索引］を参照してください。

引数

`limit`(オプション)編集リストの挿入または削除項目が指定の数を超えたら、評価を停止する。

Show Addins Dialog()

説明

アドインのステータスウィンドウを開く(［**表示**］>［**アドイン**］)。

引数

なし

Show Commands()

説明

スクリプト可能なオブジェクトと演算子をすべて表示する。引数には、All、DisplayBoxes、Scriptables、Scriptable Objects、StatTerms、Translations を指定できます。

Show Preferences(<"all">)

説明

現在の環境設定を表示する。引数が指定されていない場合、変更された環境設定を表示します。引数に "all" が指定されている場合は、すべての環境設定を表示します。

Show Properties(object)

説明

与えられたオブジェクト（*object*）が解釈できるメッセージと、その基本的な構文情報を表示する。

Sobol Quasi Random Sequence(nDim, nRow)

説明

Sobol 数列を使って最大 4000 次元の Space Filling の擬似乱数を生成する。

Socket(<STREAM | DGRAM>)

説明

ソケットを生成する。

戻り値

生成されたソケット

引数

STREAM | DGRAM（オプション）ソケットの種類がストリームかデータグラムかを指定する。引数が指定されていない場合、ストリームソケットが生成される

Speak(text, <wait(Boolean)>)

説明

システムの音声出力機能を呼び出し、テキストを読み上げる。Wait がオンになっている場合、次のスクリプトは読み上げが終了してから実行されます。

Status Msg("message")

説明

文字列（message）をステータスバーに表示する。

Subtract(a, b)

a-b-...

説明

リストされた引数の値を、左から右へと引く。どの引数も変更されません。

戻り値

差

引数

2つ以上の変数、数値、または行列。

ノート

2つ以上の引数が使えます。引数が1つまたはなしの場合は、エラーが出ます。

次も参照

『スクリプトガイド』の「データ構造」章

Unregister Addin("unique_id")

説明

登録されたアドインの登録を解除（削除）する。

引数

unique_id 登録を解除するアドインの一意の識別子を示す引用符付き文字列。

Web(string, <JMP Window>)

説明

string で指定したURLを、システムにインストールされているデフォルトのWebブラウザで開く。Microsoft Windows版では、第2引数にJMP Windowを指定すると、HTMLをJMPアプリケーション内で開きます。

URL内の http:// 接頭辞は省略することも可能です。

例

```
url = "www.jmp.com";
Web( url );

Web( "www.jmp.com" );
Web( "www.jmp.com", JMP Window );
```

XML Attr("attr name")

説明

Parse XMLコマンドの評価において、XML引数の文字列を抽出する。

XML Decode("xml")

説明

XML 内の記号を通常のテキストにデコードする。たとえば、& を &、< を < に変更します。

引数

xml XML を含んだ引用符付き文字列

XML Encode("text")

説明

XML に埋め込むテキストを準備する。たとえば、& を &、< を < に変更します。

引数

xml プレーンテキストを含んだ引用符付き文字列

XML Text()

説明

Parse XML コマンドの評価において、XML タグの本体 (ボディ) の文字列を抽出する。

第3章

JSLメッセージ

オブジェクトおよびディスプレイボックスのメッセージの概要

ここでは、JMPの一般的なオブジェクトメッセージについて簡単に説明しています。オブジェクトメッセージの詳細については、[**ヘルプ**] > [**スクリプトの索引**] で表示される「スクリプトの索引」を参照してください。

プラットフォームメッセージの詳細については、『スクリプトガイド』の「プラットフォームのスクリプト」章を参照してください。

アルファシェイプ

以下のメッセージで、ashapeはアルファシェイプまたはアルファシェイプへの参照を表します。

ashape <<Get Alpha

現在のアルファ値を戻す。

ashape <<Set Alpha(alpha)

アルファ値を設定し、アルファシェイプによる三角分割を再計算する。

ashape <<Get Tri Alpha

各三角形のアルファ値を戻す。

連想配列

以下のメッセージで、mapは連想配列または連想配列への参照を表します。

map<<First

map 内の最初のキーを戻す。*map* にキーがない場合は、Empty() を戻します。キーは辞書式の順序で戻されます。

map<<Get Contents

map 内におけるすべてのキーと値のペアを、リストで戻す。

map<<Get Keys

map 内のすべてのキーを、リストで戻す。

map<<Get Default Value()

存在しないキーで参照した場合の値を設定する。値が設定されていない場合は、Empty() を戻します。

map<<Get Value(key)

map 内の *key* に対応する値を戻す。

map<<Get Values(<keylist>)

引数が指定されない場合は、*map* 内のすべての値を、リストで戻す。

キーがリストで引数に指定された場合、それらのキーに対応する値を、リストで戻します。

map<<Insert(key, value)

map にキー（*key*）を挿入し、それに値（*value*）を割り当てる。*map* にすでに *key* がある場合は、新しい *value* で置き換えられます。このメッセージは、関数 Insert Into と等価です。

map<<Next(key)

map 内の指定のキー（*key*）の次のキーを戻す。*map* にキーがない場合は、Empty() を戻します。キーは辞書式の順序で戻されます。

map<<Remove(key)

map から、キー（*key*）とその値（*value*）を削除する。このメッセージは、関数 Remove From と等価です。

map<<Set Default Value(v)

存在しないキーで参照した場合の値を設定する。存在しないキーにはすべて、デフォルトでこの値が割り当てられます。

データテーブル

dt<<Add Column Properties(property argument, ...)

指定のプロパティを、選択された列に追加する。

dt<<Add Multiple Columns("prefix", n, position, attributes)

データテーブル（*dt*）内の指定された位置（*position*）に *n* 個の列を追加する。

dt<<Add Rows(count, <rownum>)

dt<<Add Rows(assignment list)

指定された数の行をデータテーブルの末尾、または指定された行番号（*rownum*）の場所に追加する。

dt<<Add Scripts to Table(script, ...)

dt<<Add Properties to Table(script, ...)

スクリプト（script）をデータテーブルに追加する。

dt<<Add to R

dt<<Anonymize(<columns(list)>, <Output Table Name(string)>);

データ、一部の列プロパティ、およびテーブルスクリプトから識別可能な値を削除する。データテーブル全体または選択された列に適用されます。

dt<<Begin Data Update

データテーブルのセルをすばやく更新できるように、表示の更新をオフにする。表示の更新をオンに戻すには、End Data Update を使います。

ノート

Begin Data Update を指定しても、その他のテーブル操作により再描画が行われる場合があります。たとえば、列を削除・追加した場合は、その操作に応じてデータテーブルの表示を更新してから、データの更新を始めます。

dt<<Clear Column Selection

選択されているすべての列の選択を解除する。

dt<<Clear Row States

有効な行の属性をすべてクリアする。

dt<<Clear Select

現在の選択をクリアする。

dt<<Clone Formula Column(column, n, Substitute Column Reference(column1, {list}))

n 個の新しい計算式列を作成し、*column1* への参照を *list* の列に置き換えて元の *column* の計算式に挿入する。

dt<<Close Data Grid(Boolean)

データテーブルグリッドを閉じる。

dt<<Close Side Panels(Boolean)

データテーブルのサイドパネルを閉じる。

dt<<Color or Mark by Column(col, <optional arguments>)

dt<<Color by Column(col, <optional arguments>)

機能

データテーブルの列の値に従って色またはマーカーを割り当てる。オプションの引数が指定されていない場合は、行の属性が使用されます。

引数

`Color Number(n)` 引用符付きで指定された JMP カラーを使用する。

`Color Theme("theme")` 引用符付きで指定されたカラーテーマを使用する。

`Marker Theme("named argument")` 引用符付きで指定されたマーカーテーマを使用する。指定できるテーマは、`"standard"`（標準）、`"hollow"`（中抜き）、`"paired"`（ペア）、`"classic"`（クラシック）、および、`"alphanumeric"`（英数字）。

`Continuous Scale`（`Color by Column` のみ）指定の列の値に従って、グラデーションになった色を割り当てる。

`Reverse Scale` 使用中の色を反転する。

`Excluded Rows` 除外されている列に行の属性を適用する。

`Make Window with Legend` 凡例のウィンドウを別に作成する。

dt<<Color Rows by Row State

行の属性による色の割り当てに基づき、データテーブルグリッド内の行に色をつける。行の色を無効にするには、再度このメッセージを送ります。

dt<<Compress Selected Columns({column1, ...})

リストされた列を可能な限り小さく圧縮する。文字データの列の場合、水準が 255 個未満であれば 1 バイトに圧縮できます。数字データの列の場合、数値が -127 ～ 127 であれば 1 バイトに圧縮できます。

dt<<Concatenate(dt2, ..., Keep Formulas, Output Table Name("name"))

データテーブル（*dt*）とデータテーブル 2（*dt2*）を連結し、新しいデータテーブル（*"name"*）を作成する。

dt<<Copy Table Script

データテーブルを再作成するためのスクリプトを、別の場所に貼り付けられるようにクリップボードにコピーする。

dt<<Data Filter(<Mode(...)>, <Add Filter (...)>)

データフィルタを作成する。引数が指定されていない場合は、「フィルタ列の追加」ウィンドウが表示されます。

Mode() の引数は、Select()、Show()、および Include() で、すべてブール値をとります。Select のデフォルトは真（1）、Show と Include のデフォルトは偽（0）です。

Add Filter() の引数は、Columns() と Where() です。Columns() は、カンマで区切った 1 つまたは複数の列名をとります。フィルタを定義する場合は、1 つまたは複数の Where 節を追加します。

この他にもいくつかの引数があります。詳細については、『JMP の使用法』の「JMP のレポート」章および『スクリプトガイド』の「データテーブル」章を参照してください。

dt<<Get Header Height

列見出しの高さを戻す（単位はピクセル）。

dt<<Data View(<"options">)

新しいウィンドウにデータテーブルを複製する。次の引用符付き引数のいずれか 1 つを指定した場合は、該当する行だけが新しいデータテーブルに含まれます。

excluded 引用符付き。新しいデータテーブルに、元のデータテーブルで除外されている行だけが含まれる。

labeled | labelled 引用符付き。新しいデータテーブルに、元のデータテーブルでラベルありとマークされている行だけが含まれる。

hidden 引用符付き。新しいデータテーブルに、元のデータテーブルで表示しないとマークされている行だけが含まれる。

selected 引用符付き。新しいデータテーブルに、元のデータテーブルで選択されている行だけが含まれる。

dt<<Delete Columns(col, col2, ...)

dt<<Delete Column

データテーブル（*dt*）から列（複数可）を削除する。col には、削除する列を指定します。引数がない場合、選択されている列があれば、それを削除します。Delete Column も同じです。

dt<<Delete Rows(<n>)

dt<<Delete Rows({n, o, p, ...})

dt<<Delete Rows({n::q})

現在選択されている行、または指定された行を削除する。

dt<<Delete Table Property("name")

テーブルのプロパティ（「スクリプト」など）を削除する。

dt<<Delete Table Variable("name")

テーブルの変数を削除する。

dt<<End Data Update

Begin Data Update メッセージの後で、表示の更新を再開する。これらのコマンドは、変更箇所が多数ある場合に、データテーブルをすばやく更新するために使われます。表示の更新をオフにすると更新速度が上がります。

dt<<Exclude

dt<<Unexclude

データテーブル（*dt*）内で選択された行を「除外する」から「除外しない」、または**その逆**に切り替える。

dt<<Get All Columns As Matrix

データテーブル（*dt*）のすべての列の値を行列で戻す。文字型の列には、水準に従って 1 から順に番号が付けられます。

dt<<Get As Matrix(<list of columns by name>, <list of columns by number>, <column range>)

データテーブル（*dt*）の数値列の値を行列で戻す。デフォルトでは、すべての数値列が出力されます。

例

```
dt1 = Open( "$SAMPLE_DATA/Big Class.jmp" );
colnames = dt1 << Get As Matrix(); // すべての数値列を戻す
Show( colnames );
  colnames =
  [ 12 59 95,
    12 61 123,
    12 55 74,...]
colnums = dt1 << Get As Matrix( {4, 5} ); // 4 列目と 5 列目を戻す
Show( colnums );
  colnums = [ 59 95, 61 123, 55 74, 66 145, 52 64, 60 84, 61 128, ...]

dt2 = Open( "$SAMPLE_DATA/Probe.jmp" );
colrange = dt2 << Get As Matrix( 10::22); // 10~22 列目を戻す
Show( colrange );
  colrange =
  [ -0.08818069845438 0.711340010166168 1.85904002189636 0.396923005580902
    4.50656986236572 7.86504983901978 1.53891003131866 -2.76178002357483
    0.0711032971739769 5.75577020645142 -3.62023997306824 -0.971698999404907
    -0.0525696985423565, ...]
```

dt<<Get As Report

データテーブルをレポートにして戻す。

例

次のスクリプトは、Big Class.jmpをレポートにして、分布とともに1つのウィンドウ内に表示します。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dtRpt = dt << Get As Report;
distRpt = V List Box(
   dt << Distribution(
      Continuous Distribution( Column( :Name("体重(ポンド)")) ),
      Nominal Distribution( Column( :年齢 ) )
   )
);
New Window( "例", H List Box( dtRpt, distRpt ) );
```

dt<<Get Column Names("arguments")

データテーブルの列名をリストにして戻す。*"arguments"*によって、取得される名前が次のように制限されます。

引数に"Numeric"、"Character"、"Rowstate"、"Continuous"、"Ordinal"、および"Nominal"を指定すると、指定したタイプの列だけが取り出されます。複数の引数を指定することもできます。

"String"を指定すると、列の参照リストではなく、文字列のリストを戻します。

dt<<Get Column Reference(list or matrix of col names)

リスト内または行列内の文字列の列参照を戻す。リストまたは行列が未指定の場合は、すべての列名が戻されます。

dt<<Get Display Width

列の表示幅（単位はピクセル）を戻す。

dt<<Get Excluded Rows

データテーブルで現在除外されている行を戻す。

dt<<Get Hidden Rows

データテーブルで現在非表示の行を戻す。

dt<<Get Labeled Columns

dt<<Get Labelled Columns

データテーブルで現在ラベルのついている列を戻す。

例

「PopAgeGroup.jmp」では、「**国**」列と「**年**」列にラベルがついています。次のスクリプトは、ラベルのついた列の名前のリストを戻します。

```
dt = Open( "$SAMPLE_DATA/PopAgeGroup.jmp" );
dt << Get Labeled Columns;
  {:Country, :Year}
```

---

dt<<Get Labeled Rows

dt<<Get Labelled Rows

データテーブルで現在ラベルのついている行を戻す。

---

dt<<Get Name

テーブルの名前（*name*）を戻す。

---

dt<<Get Path

JMP のデータテーブルの絶対パスを戻す。読み込んだ後にまだ保存していないデータの場合は、絶対パスは戻されません。

---

dt<<Get Property("name")

name で指定されたテーブルプロパティの*スクリプト*を戻す。

---

dt<<Get Row Change Function

行が選択されたときに評価する式を戻す。

---

dt<<Get Row ID Width

行番号の領域の表示幅（単位はピクセル）を戻す。

---

dt<<Get Row States

データテーブルまたはデータフィルタの各行の行属性を含むベクトルを戻す。

---

dt<<Get Rows Where(WHERE clause)

指定の where 条件とマッチするデータテーブルの行を戻す。以下はその例です。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Get Rows Where( :sex == "M" );
dt << Get Rows Where( :sex == "M" & :age < 15 );
```

dt<<Get Script

データテーブルを再作成するためのスクリプトを式で戻す。

dt<<Get Selected Columns(<"string">)

選択されている列のリストを列参照として戻す。選択された列を文字列で戻すには *"string"* 引数を使用します。

例

```
dt << Get Selected Columns();
  {:age, :sex, :height}
dt << Get Selected Columns( "string" );
  {"age", "sex", "height"}
```

dt<<Get Selected Rows()

選択されている行を戻す。

dt<<Get Table Script Names()

データテーブルのすべてのスクリプト名およびプロパティ名のリストを戻す。

dt<<Get Table Variable("name")

テーブル変数 *name* に含まれている値を戻す。

dt<<Get Table Variable Names()

すべてのテーブル変数の変数名を、リストで戻す。

dt<<Go To Row(n)

データテーブル（*dt*）の番号 *n* の行を選択する。

dt<<Group Columns({col1, col1, ...})

dt<<Group Columns("group name", col, n)

指定した列を、指定した名前でグループ化する。グループ化する列をリストの形で指定するか、最初の列の名前と列数という形で指定します。後者の場合、数字 *n* は、その列とそれに続く *n*-1 個の列をグループ化することを意味します。

dt<<Hide

dt<<Unhide

データテーブル（*dt*）内で選択された行を「表示しない」から「表示する」、または**その逆**に切り替える。

dt<<Invert Row Selection

現在選択されていない行を選択し、選択されている行の選択を解除する。

dt<<Is Dirty

テーブルが保存された状態から変更されている場合は1、そうでない場合は0を戻す。

dt<<JMP Query Builder

1つまたは複数のデータテーブルに対するクエリーを作成する。

dt<<Join(With(dataTable), <Private>, <Invisible>, Select(columns), Select With(columns), (By Matching Columns(col1=col2, ...)|Cartesian|By Row Number), <Merge Same Name Columns>, <Match Flag>, <Copy Formula(Boolean)>, <Suppress Formula Evaluation(Boolean)>, Update, <Drop Multiples(Boolean, Boolean)>, <Include Non Matches(Boolean, Boolean)>, <Preserve Main Table Order>, <Output Table Name("name"))>

説明

データテーブル（*dt* および *table*）を横に並べて結合する。テーブルの結合の詳細については、『JMPの使用法』の「データの再構成」章を参照してください。

戻り値

データテーブル

引数

Private（オプション）結果のデータテーブルを開かない。

Invisible（オプション）データテーブルを開き、非表示にする。このテーブルは、JMPのホームウィンドウの「ウィンドウリスト」または［ウィンドウ］>［再表示］のリストに表示されます。

With(dataTable) 2番目のデータテーブル。

Select(columns) メインのデータテーブルから、出力テーブルに追加する列を選択する。（オプション）

Select With(columns) 2番目のデータテーブルから、出力テーブルに追加する列を選択する。

By Matching Columns(col1=col2, ...)|Cartesian|By Row Number テーブルを結合する方法を指定する。By Matching Columns(col1=col2, ...) は、col1 と col2 の値を比較し、一致した値を結合します。Cartesian（直積）で結合すると、2つの元のデータテーブルの行の可能な組合せがすべて含まれた新しいテーブルが作成されます。最初のテーブルのデータを2番目のテーブルのデータと組み合わせて、値のすべての組合せがセットで表示されます。By Row Number は、2つのテーブルを横に並べて結合します。By Matching Columns がデフォルトです。

Merge Same Name Columns（オプション）同じ名前の列をマージする。デフォルトではオフになっています。

Match Flag（オプション）デフォルトではオフ。対応する列の値で結合する場合に、結合後のデータテーブルに「対応フラグ」列を作成しません。

Copy Formula(Boolean for main table, Boolean for secondary table)（オプション）結合元のテーブルから計算式をコピーする。デフォルトではオンです。

Suppress Formula Evaluation()（オプション）新しいテーブルの作成中に、列の計算式を再評価しない。デフォルトではオンです。メインのテーブルのみで再評価しない場合は、Suppress Main Table Formula Evaluation を使用します。2 番目のテーブルのみで再評価しない場合は、Suppress Second Table Formula Evaluation を使用します。

Update（オプション）メインのテーブルのデータを、2 番目のテーブルの対応するデータで置き換える。

Drop Multiples(Boolean for main table, Boolean for secondary table)（オプション）最初に検出された一致する行だけを新しいデータテーブルに書き込む。［重複する行を削除］のチェックボックスを両方ともオフにしておくと、対応する列の各値の直積となります。デフォルトではオフです。

Include Non Matches(Boolean for main table, Boolean for secondary table)（オプション）一致しない列を含める。デフォルトではオフです。

Preserve Main Table Order（オプション）対応する列を基準に並べ替えるのではなく、主テーブルの順序を結合後のテーブルでも維持する。デフォルトではオンです。

Output Table Name("name"))（オプション）出力テーブルの名前。

dt<<Journal

データテーブルからジャーナルを作成する。データグリッドのみが含まれ、ノート、変数、またはスクリプトは含まれません。

dt<<Journal Link

現在のジャーナルに、データテーブルへのリンクを追加する。ジャーナルが存在しない場合は、新しいジャーナルが作成されます。

ノート

このメッセージでジャーナルにデータテーブルへのリンクを作成するには、Open() を使用します。データテーブルが保存済みの場合は、Open() には正しいパスとファイル名が含まれますが、データテーブルがまだ保存されていない場合は、Open() には不正確なファイル名のみが含まれます。

これに対処するためには、Journal Link メッセージをデータテーブルオブジェクトに送る前に、データテーブルを保存します。以下に例を示します。

```
dt1 = New Table( "テスト",
   Add Rows( 3 ),
   New Column( "年齢", Numeric, Continuous, Format( "Best", 12 ), Set Values( [8,
   9, 10] ) )
);

// ジャーナルとリンクする前にテーブルを保存する
If( Host is( Windows ),
   dt1 << Save( "$DESKTOP/Test.jmp" ),
   dt1 << Save( "$DESKTOP/Test.jmp" )
);
```

```
dt1 << Journal Link;
```

dt<<Label

dt<<Unlabel

データテーブル（*dt*）内の選択された行を「ラベルあり」から「ラベルなし」、または**その逆**に切り替える。

dt<<Last Modified

データテーブルが最後に保存された日付を戻す。

dt<<Layout

データテーブルからレイアウトウィンドウを作成する。データグリッドのみが含まれ、ノート、変数、またはスクリプトは含まれません。

dt<<Lock Data Table

データテーブルをロックして、データや列プロパティが追加または変更されないようにする。

dt<<Make Indicator Columns(<Append Column Name(*Boolean*)>, <Include Missing(*Boolean*)>);

指定したカテゴリカル列について、0 または 1 の値をとる指示変数の列を作成する。

例

次の例では、「性別」列の指示変数列を作成します。Append Column Name を指定すると、作成される列名は「性別 _F」と「性別 _M」になります。指定しない場合は、水準値が列名となります（「**F**」と「M」）。Include Missing は、欠測値を含めます。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
dt << Make Indicator Columns(
   Columns( :性別 ),
   Append Column Name( 1 ),
   Include Missing( 1 )
);
```

dt<<Make RowState Handler

行の属性ハンドラの関数を作成する。関数の引数には、行の属性が変更された行をとる。

dt<<Make SAS Data Step

データテーブルと同じデータを作成するための SAS データステップを生成する。

dt<<Make SAS Data Step Window

データテーブルと同じデータを作成するためのSASデータステップを生成し、それをSASスクリプトウィンドウに表示する。

dt<<Marker by Column(col)

データテーブルの列の値に従ってマーカーを割り当てる。

dt<<Markers(n)

選択されている行にマーカー番号 *n* のマーカーを割り当てる。

dt<<Maximize Display

すべての列のサイズを再調整して、データテーブルを最適なサイズのウィンドウに拡大する。

dt<<Move Selected Columns("To First"|"To Last"|After(col))

データテーブル内で選択されている列を、指定の位置に移動する。

dt<<Move Rows("At Start"|"At End"|After(n))

データテーブル内で選択されている行を、指定の位置に移動する。

dt<<New Column("name", attributes)

データテーブル（*dt*）の最後に、指定された名前（*"name"*）で新しい列を追加する。これは関数として使用することもできます。New Column("name", attributes)

dt<<New Data Box()

ディスプレイボックスツリーに、データテーブルビューを作成する。データテーブルとレポートを1つのウィンドウに表示するのに便利です。データテーブルオブジェクトにNew Data Boxメッセージを送ると、データブラウザボックスが作成されます。

例

次のスクリプトは、データテーブルビューとレポートを作成し、1つのウィンドウに表示します。データテーブルは、データブラウザボックスに配置されます。ボックスの幅は、800ピクセルに設定されます。自動伸縮をオフにしているため、ウィンドウの右端を伸ばしても、データテーブルビューの幅は800ピクセルに保たれます。

```
dtA = Open( "$SAMPLE_DATA/Semiconductor Capability.jmp", invisible );
nw = New Window( "例",
   H List Box(
      V List Box( dtbox = dtA << New Data Box() ),
      dtA << Distribution(
         Continuous Distribution( Column( :NPN1 ) ),
```

```
            Continuous Distribution( Column( :PNP1 ) )
        )
    )
);
dtbox << Set Auto Stretching( 0, 0 ) << Set Width( 800 );
```

dt<<New Data View

データテーブルの複製を開く。2 番目のデータテーブルは元のデータテーブルと同一で、元のデータテーブルにリンクしています。そのため、どちらか一方が変更されると、他方にも反映されます。また、どちらか一方を閉じると他方も閉じられ、データテーブルへの参照はすべて削除されます。

これは、非表示のデータテーブルを表示する際に便利です。

dt<<New Script("name", script)

dt<<Set Property("name", script)

指定のスクリプト（*script*）を格納する新しいテーブルプロパティを、指定の名前（*name*）で作成する。

今後サポートされなくなる New Property() および New Table Property() の代わりに New Script() または Set Property() を使用してください。

dt<<New Table Variable("name", value)

dt<<Set Table Variable("name", value)

指定の名前（*name*）と値（*value*）で新しいテーブル変数を作成する。

dt<<Next Selected

次の選択行を表示する。次の選択行が画面の外にある場合は、データテーブルを下にスクロールします。

dt<<Original Order

データテーブル（*dt*）の列の順を保存されている順序に戻す。

dt<<Previous Selected

前の選択行を表示する。前の選択行が画面の外にある場合は、データテーブルを上にスクロールします。

dt<<Print Window(<"Show Dialog">)

ウィンドウを印刷する。オプションの引数 "Show Dialog" が指定されている場合は、印刷ウィンドウが表示されます。そうでない場合は、ウィンドウはデフォルトのプリンタに現在の設定を使って印刷され、印刷ウィンドウは表示されません。

dt<<Reorder By Data Type

データテーブル（*dt*）の列を、データタイプによって行の属性、文字、数値の順に並べ替える。

dt<<Reorder By Modeling Type

データテーブル（*dt*）の列を、尺度によって連続尺度、順序尺度、名義尺度の順に並べ替える。

dt<<Reorder By Name

データテーブル（*dt*）の列を名前の昇順に並べ替える。

dt<<Rerun Formulas

テーブル変数を含んだ、列の計算式を、すべて再計算する。再計算は適切な順序で実行されます。

dt<<Reverse Order

データテーブル（*dt*）の列順を逆にする。

dt<<Revert

最後に保存されたデータテーブル（*dt*）に戻す。

dt<<Run Formulas

他の計算式を評価した後で評価するために保留されている計算式も含め、データテーブルのすべての計算式の評価を実行する。

dt<<Run Script("name")

テーブルパネルにある（*name*）プロパティの JSL スクリプトを実行する。

dt<<Save("path")

dt<<Save As("path")

指定されたパス（*"path"*）にテーブルを保存する。

サポートされている形式については、『JMP の使用法』の「データの保存と共有」章を参照してください。

dt<<Save Database("connection_information", "table_name", <Replace>)

指定の接続とテーブル名を使って、データテーブルをデータベースに保存する。Replace オプションを指定した場合は、既存のデータベースを置換します。

dt<<Save Script to Script Window

データテーブルを再作成するためのスクリプトをスクリプトウィンドウに保存する。

dt<<Select All Rows

データテーブル（dt）のすべての行を選択する。

dt<<Select Excluded

データテーブル（dt）で現在除外されている行だけを選択する。

dt<<Select Hidden

データテーブル（dt）で現在非表示になっている行だけを選択する。

dt<<Select Labeled

データテーブル（dt）で現在ラベルがついている行だけを選択する。

dt<<Select Randomly(p|n)

指定された割合（*p*）、もしくは行数（*n*）でデータテーブルの行を無作為に選択する。

dt<<Select Rows({list})

行番号のリスト（list）で指定された行を選択する。

dt<<Select Where(condition)

データテーブル（*dt*）で、条件（condition）が真になる行を選択する。

dt<<Set Dirty(Boolean)

変更されていない場合でも、データテーブルを変更済みとしてマークする。

dt<<Set Header Height(num)

列見出しの高さを指定のピクセル数に設定する。

dt<<Set Label Columns(column_1, ...)

指定の列をラベル列として割り当てる。

dt<<Set Matrix(matrix)

データテーブルに行列を挿入し、必要に応じて新しい列と行を追加する。

dt<<Set Name("name")

テーブルに名前（*"name"*）をつける。新しいデータテーブル名が文字列として戻されます。

ノート

Set Name メッセージが、新しいテーブル名の文字列を戻すように変更されました。以前のリリースでは、このメッセージは、スクリプト可能なデータテーブルオブジェクトを戻していました。この変更に伴い、JMP スクリプトを変更する必要がある場合があります。たとえば、以下のスクリプトは書き換える必要があります。

```
dt = Open( "$SAMPLE_DATA\Big Class.jmp" ) << Set Name( "テスト" );
```

dt が「テスト」にならないように、メッセージを分けます。

```
dt = Open( "$SAMPLE_DATA\Big Class.jmp" );
dt << Set Name( "テスト" );
```

これは以前のリリースと同じ結果になり、JMP の以前のバージョンと新しいバージョンの両方に対応します。

dt<<Set Property("name", script)

「dt<<New Script("name", script)」（277 ページ）を参照してください。

dt<<Set Label Columns("name", ...)

dt<<Set Label Columns

指定した列のラベルの属性をオン（ラベルあり）にする。列がリストされていないときは、ラベルの属性はオフ（ラベルなし）になります。

dt<<Set Row ID Width(expr)

行番号の領域の表示幅を expr に設定する（単位はピクセル）。

dt<<Set Row States(matrix)

データテーブル内のすべての行の属性を設定する。

dt<<Set Scroll Lock Columns("name", ...)

指定した列のスクロールをロックする。列が指定されていないときは、スクロールのロックを解除します。

dt<<Set Table Variable("name", value)

「dt<<New Table Variable("name", value)」（277 ページ）を参照してください。

dt<<Sort(By(columns), order("Descending"または"Ascending"), Output Table Name("name"))

列（*columns*）の値に従って、データテーブル（*dt*）の行を並べ替え、新しいデータテーブル（*"name"*）を作成する。

dt<<Split (Split(column), Split by(column), Group(column), <Private> | <Invisible>, <Remaining Columns(Keep All | Drop All | Select(columns))>, <Copy formula(0|1)>, <Suppress formula evaluation(0|1)>, <Sort by Column Property("Value Ordering" {"*string*", "*string*"} | "Row Order Levels")>, <Output Table ("name")>)

Split で指定した各列を Split by の列の値により複数の列に分割する。Split と Split by の引数は必須です。

dt<<Stack(Stack(columns), ID(columns), Stacked(newcol), Output Table Name("name"))

データテーブル（*dt*）内の複数列の値を 1 列（*newcol*）に積み重ねて、新しいテーブル（*"name"*）を作成する。

dt<<Subscribe("keyname"(<"client">), On Delete Columns | On Add Columns | On Add Rows | On Delete Rows | On Rename Column | On Close | On Save | On Rename (function))

そのデータテーブルへの変更に関するメッセージを取得するため、データテーブルに登録する。

dt<<Subset(Columns(columns), Rows(matrix), Linked, Table Name("name"), Copy Formula(1|0), Suppress Formula Evaluation(1|0), Sampling Rate(rate))

データテーブル（*dt*）から指定された行と列を抽出し、新しいテーブル（*"name"*）を作成する。

dt<<Summary(<Private>, <Invisible>, Group(column), Subgroup(column), <N>, <Mean(column)>, <Std Dev(column>, <Min(column)>, <Max(column)>, <Range(column)>, <Sum(column)>, <CV(column)>, Freq(<freq column>), Weight(<weight column>), Include marginal statistics, Link to original data table(0), statistics column name format("stat(column)" | "column" | "stat of column" | "column stat")

指定した *col* の要約統計量を含むデータテーブルを（オプションで、グループやサブグループごとに）作成する。

dt<<Suppress Formula Eval(Boolean)

引数が 1 の場合、データテーブル (*dt*) の計算式の自動評価をオフにする。引数が 0 の場合は、オンにする。

dt<<Text to Columns

区切り文字で区切られた文字値から、別々の文字値または指示変数の列を作成する。

例

```
dt = Open( "$SAMPLE_DATA/Consumer Preferences.jmp" );
dt << Text To Columns(
   delimiter( "," ),
   columns( :歯磨き カンマ区切り )
);
```

dt<<Transpose(Columns(columns), Rows(matrix), Output Table Name("name"))

指定の行と列を転置し、新しいテーブル（*"name"*）を作成する。

dt<<Ungroup Columns({col1, col1, ...})

リスト引数で定義された列のグループを解除する。

dt<<Unsubscribe("keyname", On Delete Columns | On Add Columns | On Add Rows | On Delete Rows | On Close | On Col Rename | All)

データテーブルへの以前の登録を解除する。

列

col<<Add Column Properties ("name", expression)

指定の式（*expression*）を持つプロパティ（*"name"*）を追加する。標準の列プロパティまたはユーザ指定のプロパティを追加できます。

col<<Add From Row States

データテーブルの各行に設定されている「行の属性」を、列に含まれている「行の属性」に追加する。

col<<Add To Row States

列に含まれている「行の属性」を、データテーブルの各行がもつ「行の属性」に追加する。

col<<Color Cells("color")

データテーブルグリッドの列のセルに色をつける。引用符付きで指定された任意の色名を使用します。色をクリアする場合は 0 を指定します。

col<<Color Cell by Value(Boolean)

「値の色」プロパティに基づいて、データテーブルグリッドの列のセルに色をつける。

col<<Copy Column Properties

列プロパティをバッファ内にコピーする。

col<<Copy From Row States

データテーブルの各行に設定されている「行の属性」を、列にコピーする。

col<<Copy to Row States

列に含まれている「行の属性」を、データテーブルの各行がもつ「行の属性」に設定する。

col<<Data Type("type")

列（*col*）のタイプ（*"type"*）を設定する。指定できるタイプは、"Numeric"（数値）、"Character"（文字）、"Rowstate"（行の属性）です。

col<<Delete Formula

列から計算式を削除する。

col<<Delete Property(name)

col<<Delete Column Property(name)

指定された名前（name）の列プロパティを削除する。

col<<Eval Formula

計算式を強制的に評価する。

col<<Exclude(0|1)

指定されたブール引数に従って、「除外する」の属性をオン／オフにする。

col<<Format("format", <width>, <decimal>, <"Use Thousands Separator">)

指定された数値表示形式（*format*）を設定する。

col<<Formula(expression)

列に計算式を設定し、その計算式を評価する。

col<<Get Column Field Width

列データの表示に適用されているフィールド幅を戻す。

col<<Get Data Type

列（*col*）のデータタイプを戻す。

col<<Get Format

列の表示形式を戻す。

col<<Get Formula

計算式を戻す。

col<<Get Input Format

その列へのデータの入力および保存に適用されている形式を戻す。

col<<Get List Check

リストチェックの定義を戻す。列にリストチェックが定義されていない場合は、そのことを示すメッセージがログに送られます。

col<<Get Lock

現在のロック設定を戻す。

col<<Get Modeling Type

列の尺度を戻す。尺度には "Continuous"（連続尺度）、"Ordinal"（順序尺度）、"Nominal"（名義尺度）があります。

col<<Get Name

列の名前を戻す。

col<<Get Property

指定のプロパティ定義を戻す。列にそのプロパティが定義されていない場合は、そのことを示すメッセージがログに送られます。

col<<Get Range Check

範囲チェックの定義を戻す。列に範囲チェックが定義されていない場合は、そのことを示すメッセージがログに送られます。

col<<Get Role

列（*col*）に事前に割り当てられている役割を戻す。

col<<Get Script

列を再現するスクリプトを戻す。

col<<Get Selected

列が選択されている場合は 1、そうでない場合は 0 を戻す。

col<<Get Value Labels

値ラベルの定義を戻す。列に値ラベルが定義されていない場合は、そのことを示すメッセージがログに送られます。

col<<Get Use Value Labels

列に値ラベルを使用するよう設定されている場合は 1、そうでない場合は 0 を戻す。

col<<Get Values

列の値を戻す。

col<<Hide(Boolean)

指定されたブール引数に従って、「表示しない」の属性をオン／オフにする。

col<<Ignore Errors

列内の計算式を評価中にエラーが発生した場合、セルの値を欠測値に設定する。

col<<Input Format("format")

その列へのデータの入力および保存に適用されている形式を設定する。引数は、任意の JMP 形式の名前（日付値の列なら "ddmmyyyy" など）です。

date_col<<Is Transformed On SAS Export

データを SAS に書き出して SAS データセットを作成する際、日付列のデータが変更される場合は真を戻す。

col<<Label(Boolean)

指定されたブール引数に従って、ラベル属性をオン／オフにする。

col<<Lock(Boolean)

col<<Set Lock(Boolean)

指定されたブール引数に従って、ロック属性をオン／オフにする。

col<<Preselect Role("role")

列（col）の役割を事前に設定しておく。選択肢は、"Y"、"X"、"Weight"（重み）、"Freq"（度数）、"None"（なし）または "No Role"（役割なし）です。

col<<Set Scroll Locked(Boolean)

指定されたブール引数に従って、スクロールロック属性をオン／オフにする。

col<<Set Display Width(n)

列の表示幅を n に設定する（単位はピクセル）。

col<<Set Each Value(n)

列のすべての値を *n* に設定する。

col<<Set Field Width(n)

列のフィールド幅を *n* に設定する。

col<<Set Modeling Type("type")

変数の尺度（type）を設定する。選択肢は、"Continuous"（連続尺度）、"Ordinal"（順序尺度）、Nominal"（名義尺度）です。

col<<Set Name("name")

列の名前を設定する。

col<<Set Property ("name", {argument list})

指定された式（*expression*）をプロパティ（*"name"*）に設定する。標準の列プロパティまたはユーザ指定のプロパティを設定できます。

col<<Set Selected(Boolean)

列が選択されるように（true または 1）、または選択されないように（false または 0）設定する。

col<<Set Values([matrix] or {list})

col<<Values([matrix] or {list})

行列（数値変数の場合）またはリスト（文字変数の場合）の値を列の値として設定する。

col<<Suppress Eval(Boolean)

列の計算式に対する自動評価を、オフにする。

行

row<<Colors(n)

選択されている行に番号 *n* の色を割り当てる。

row<<Exclude(0|1)

row<<Unexclude(0|1)

指定されたブール引数に従って、行の「除外する」の属性をオン／オフにする。引数を省略すると、現在の状態と逆に設定されます。

row<<Hide(0|1)

row<<Unhide(0|1)

指定されたブール引数に従って、「表示しない」の属性をオン／オフにする。引数を省略すると、現在の状態と逆に設定されます。

row<<Label(0|1)

row<<Unlabel(0|1)

指定されたブール引数に従って、ラベル属性をオン／オフにする。引数を省略すると、現在の状態と逆に設定されます。

row<<Markers("marker")

選択されている行に "*marker*" で指定されたマーカーを割り当てる。

row<<Next Selected

次の選択行を点滅させる。

row<<Previous Selected

前の選択行を点滅させる。

row<<Row Editor

選択されている行の編集ウィンドウを開く。

データフィルタ

dtf<<Add Filter(columns(column <,col>), <Where(clause)>)

1つまたは複数のフィルタ列をORの条件で追加する。

dtf<<Auto Clear(0|1)

新しい選択項目を設定する前に、現在選択されている行をすべてクリアする。

dtf<<Clear

現在選択されている行をクリアする。

dtf<<Close

データフィルタウィンドウを閉じる。

dtf<<Columns(col1, col2, ...)

データフィルタで使用する列を設定する。

dtf<<Data Table Window

データフィルタウィンドウが使用しているデータテーブルを表示する。

dtf<<Delete All

設定されているすべてのフィルタを削除する。

dtf<<Delete(col1, col2, ...

指定の列をデータフィルタから削除する。

dtf<<Display(col, <Size(x, y)>, "Blocks Display"|"List Display"|"Single Category Display"|"Checkbox Display")

指定のカテゴリカル列の水準を、どのようにフィルタに表示するかを設定する。指定する列はカテゴリカルでなければなりません。

dtf<<Get Script

データフィルタのスクリプトをテキストとしてログに戻す。

例

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
df = dt << Data Filter(
     Add Filter( Columns( :年齢 , :性別 ), Where( :年齢 == 12 ) )
);

txt = df << Get Script;
Show( txt );
```

dtf<<Local Data Filter

指定のウィンドウにデータフィルタを組み込む。

dtf<<Location(x, y)

データフィルタウィンドウを指定の場所に移動する。x と y はピクセル単位で指定します。0,0 はディスプレイの左上隅を意味します。

dtf<<Make Subset

データフィルタで選択されている行を含む、新しいサブセットデータテーブルを作成する。

dtf<<Match(Filter Columns(:col1, :col2, ...), Where(WHERE clause)

各列のフィルタ条件を設定する。WHERE 節は、リストされたすべての列で使用されます。列ごとに異なる WHERE 節を使用するには、Match メッセージを各列に個別に送ります。

dtf<<Mode(Select(0|1)|Show(0|1)|Include(0|1))

データフィルタを使って行が選択されたときのアクションまたはモードを設定する。

dtf<<Save and restore current row states

データテーブルの現在の「行の属性」を保存しておき、その後でデータフィルタを閉じたときに、それらの属性を復元する。

dtf<<Show Columns Selector(0|1)

フィルタの完了後、列セレクタを表示または非表示にする。

dtf<<to Clipboard

データフィルタの現在の状態から WHERE 節を作成し、クリップボードに置いて、別の場所に貼り付けることができるようにする。

dtf<<to Data Table

データフィルタの現在の状態から WHERE 節を作成し、プロパティとしてデータテーブルに保存する。

dtf<<to Journal

データフィルタの現在の状態から WHERE 節を作成し、現在のジャーナルに追加する。現在のジャーナルが存在しない場合、新しいジャーナルが開き、WHERE 節はそれに追加されます。

dtf<<to Row State Column

計算式が WHERE 節である行の属性列を作成する。

dtf<<to Script Window

データフィルタの現在の状態から WHERE 節を作成し、現在のスクリプトウィンドウに追加する。現在のスクリプトウィンドウが存在しない場合、新しいスクリプトウィンドウが開き、WHERE 節はそれに追加されます。

dtf<<Use Floating Window(0|1)

データフィルタウィンドウを、関連するデータテーブルの上に配置するか、通常のウィンドウとして表示するかを設定する。

dtf<<Where(WHERE clause)

行選択の条件を設定する。

データベース

dt<<Save Database("connectInfo", "TableName")

データテーブルをデータベースに保存する。

データフィード

feed<<Close

データフィードオブジェクトとそのウィンドウを閉じる。

feed<<Connect(portSettings)

（Windows のみ）デバイスに接続するために、ポートの設定を行う。

feed<<Disconnect

（Windows のみ）データフィードオブジェクトをアクティブにしたまま、データフィードのキューとデバイスとの接続を切断する。

feed<<Get Line

データフィードのキューの中から 1 ラインのデータを戻し、削除する。

feed<<Get Lines

データフィードのキューのデータすべてをリストにして戻し、削除する。

feed<<Queue Line(string)

データフィードのキューの最後へ 1 ラインのデータ送る。

feed<<Restart

データラインへの処理を再開する。

feed<<Set Script(script)

データラインが届くたびに実行されるスクリプト（*script*）を設定する

feed<<Stop

データラインへの処理を停止する。

ディスプレイボックス

他の例については、『スクリプトガイド』の「表示ツリー」章および JMP の［スクリプトの索引］を参照してください。

すべてのディスプレイボックス

db<<Add Text Annotation(Text("string"), Text Box(x1, y1, x2, y2))

文字列を含むテキスト注釈ボックスを描画する。Text Box 引数によって、テキスト注釈ボックスを描く位置を、ディスプレイボックスにおける相対的な位置（左上から右下）で指定します。

db<<Append(db2)

db2 を表示ツリーの *db* の後ろに追加する。

db<<Child

ボックスの子を戻す。

db<<Class Name

ディスプレイボックスの表示クラス名を戻す。

db<<Clone Box

ディスプレイボックスの新しいコピーを作成する。

db<<Close Window

ディスプレイボックスが表示されているウィンドウを閉じる。

db<<Copy Picture

ディスプレイボックスのイメージをクリップボード上に置く。

db<<Delete

ディスプレイボックスを削除する。

db<<Enable(Boolean)

ディスプレイボックスを操作可能にするかどうかを指定する。0 の場合はディスプレイボックスを操作できず、1 だとディスプレイボックスを操作できます。

db<<Get HTML

ディスプレイボックスの HTML ソースを含んだ文字列を戻す。

db<<Get Journal

ディスプレイボックスのジャーナルソースを含んだ文字列を戻す。

db<<Get Menu Item State(index)

指定されたポップアップメニュー項目の状態を戻す。状態は、通常（0）、選択されている（1）、または選択不可（-1）のいずれか。

db<<Get Menu Items

ボタンがクリックされたときに呼び出されるポップアップメニューの項目を戻す。サブメニューについては、<<Get Submenu(index) を参照してください。

db<<Get Menu Script

オブジェクトに指定されているメニュースクリプトを戻す。

db<<Get Page Setup()

ページ設定の設定内容を戻す。

例

下の例では、新しいウィンドウを作成し、ページ設定の設定内容を戻します。

```
w = New Window( "Window",
   Text Box( "Page Setup Test" )
);
w << Get Page Setup();
```

メッセージの結果は次のとおりです。

```
{Margins( {0.75, 0.75, 0.75, 0.75} ), Scale( 1 ), Portrait( 1 ),
Paper Size( "Letter" )}
```

db<<Get Picture(<Scale(*n*)>)

ディスプレイボックス（*db*）をイメージオブジェクトとして取得する。スケールは、元のイメージサイズが基準となります。たとえば、Scale(2) を指定すると、イメージオブジェクトのサイズが 2 倍になります。

db<<Get RTF

ディスプレイボックスの RTF ソースを含んだ文字列を戻す。

db<<Get Script

ディスプレイボックスを再作成するためのスクリプトを戻す。

db<<Get Size

{ x, y } または { h, v } をピクセルで戻す。

```
xy = DisplayBox << Get Size;
```

x と y をピクセルで戻す。

```
{ x, y } = DisplayBox << Get Size;
```

db<<Get Submenu (index)

指定されたメニュー項目の下位にあるサブメニュー項目の数を戻す。

例

下の例は、"A"、"B"、"C" というメニュー項目があるメニューを作成します。"A" にはサブメニュー項目 "A1" と A2" を、"B" にはサブメニュー項目 "B1"、"B2"、"B3" を設定しています。<<Get

Submenu(inc) は、インデックスを指定した各メニュー項目の下位にあるサブメニュー項目の数を戻します。

```
New Window( "Title",
obj = Outline Box( "title" ) );
submenus = { };
obj << Set Menu Script(
   {"A", "", "A1", Print( "A1" ), "A2", Print( "A2" ),
   "B", "", "B1", Print( "B1" ), "B2", Print( "B2" ), "B3", Print( "B3" ),
   "C", Print( "C" )}
);
obj << Set Submenu( 1, 2 ); // メニューA、サブメニューにA1とA2の2項目
obj << Set Submenu( 4, 3 ); // メニューB、サブメニューにB1、B2、B3の3項目
For( inc = 1, inc <= N Items( Words( obj << Get Menu Script, "," ) ), inc++,
   Insert Into( submenus, obj << Get Submenu( inc ) );
);
submenus;
  {2, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

このログ出力は、インデックス（1）にサブメニュー項目が2つ、インデックス（3）にサブメニュー項目が3つあることを示しています。

db<<Get Text

ディスプレイボックスのテキストを含んだ文字列を戻す。

db<<Horizontal Alignment("position")

ディスプレイボックスが入れ子になっている場合、子ディスプレイボックスの位置を *position* で指定する。デフォルト値は left（左揃え）です。center（中央揃え）、または right（右揃え）を指定することもできます。

例

```
New Window( "例",
   Outline Box( "親ディスプレイボックス",
      Button Box( "OK", <<Horizontal Alignment( "Center" ) )
   )
);
```

db<<Inval

ウィンドウ内のディスプレイボックス領域を無効にする。ウィンドウは、次回、オペレーティングシステムによってウィンドウが更新される際（たとえば、ユーザがディスプレイボックスのサイズを変更したとき）に、更新されます。

ノート

Wait(0) を使う代わりに、新しいメッセージ <<Update Window の使用を検討してください。Wait(*n*) を使う場合は、*n* をどの程度大きい値にするか決めておく必要がある点が問題です。

ディスプレイボックスの多くのメッセージは（<<Set Text など）、ボックスを自動的に無効としてマークするため、<<Inval メッセージは通常不要です。スライダと JSL コールバックを使うインタラクティブなスクリプトでは、ディスプレイの各所をスライダと同期させるために、<<Update Window が必要になる場合があります。

db<<Is Enabled

コントロールが有効になっているかどうかの状態を戻す。このメッセージは、Busy Light Box()、Button Box()、Calendar Box()、Check Box()、Col List Box()、Combo Box()、Completion Box()、Filter Col Selector()、gtext()、List Box()、Number Edit Box()、Popup Box()、Radio Box()、Range Slider Box()、Slider Box()、Spin Box()、Text Edit Box()、Tree Box()、Tree Map Box()、Tree Map Seg() でサポートされています。

db<<Journal

ディスプレイボックスをジャーナルに追加する。

db<<Journal Window

ディスプレイボックスが表示されているウィンドウ全体をジャーナルに追加する。Journal と比較してください。

db<<Move Window(x, y)

ウィンドウをスクリーン上の (x, y) の位置に移動する。

db<<Page Break

ディスプレイボックスの前にページ区切りを挿入する。

db<<Parent

このディスプレイボックスの親を戻す。

db<<Prepend(db2)

db2 を表示ツリーの *db* の前に追加する。

db<<Reshow

ウィンドウ内のディスプレイボックスの領域を無効にし、ウィンドウの内容を更新する。

db<<Save Capture("path", <"format">, <Add Sibling(n)>)

ディスプレイボックスをスクリーンキャプチャーし、指定のパス（*"path"*）に、指定の形式（*"format"*）のグラフィックで保存する。オプションの引数（Add Sibling）は、取り込む兄弟ディスプ

レイボックスの数を指定します。デフォルト値は1で、この場合、指定のディスプレイボックスだけを取り込みます。ディスプレイボックスが、別のウィンドウに隠されていたり、スクロールしないと表示されなくなっていたりする時には注意が必要です。ディスプレイボックスが画面に表示されていない場合は、期待どおりの結果にはならないかもしれません。

db<<Save HTML("" | "path", "format")

HTML ソースファイルを保存する。指定された形式（*"format"*）のグラフィックファイルを格納したフォルダも一緒に保存されます。

db<<Save Interactive HTML("path")

インタラクティブ HTML 5 機能を使った Web ページとしてディスプレイボックスを保存する。こうすれば、JMP を使用していないユーザーでも、データを見ることができます。

例

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate(y(:Name("体重（ポンド）")), x(:Name("身長（インチ）")));
rbiv = (biv << Report);
rbiv << Save Interactive HTML( "$DOCUMENTS/MyInteractiveHTML.htm" );
```

db<<Save Journal("" | "path")

ディスプレイボックスのジャーナルソースを保存する。

db<<Save MSWord("pathname", Native)

ディスプレイボックスを Microsoft Word 文書として保存する （Windows のみ）。

db<<Save Picture("" | "pathname", "format")

ディスプレイボックスのイメージを保存する。

ノート

有効な形式には、PNG、GIF、JPG、JPEG、EMF があります。

Windows の場合は、［Windows のみ］環境設定で解像度（DPI）を指定できます。または、次のスクリプトを実行できます。

```
Pref( Save Image DPI( <number> ) );
```

Macintosh の場合は、オペレーティングシステムにより DPI が決まります。

```
db<<Save Presentation("path", <template("path")>,
<(Insert("Begin"|"End"|N) | Replace("Begin"|"End"|N) | "Append")>,
<Outline
Titles("None"|"Hide"|"TopLeft"|"TopRight"|"BottomLeft"|"BottomRight"
)>, <format>)
```

Microsoft PowerPoint ファイルにディスプレイボックスを保存する。このファイルは、任意のプレゼンテーションソフトウェアプログラムで開くことができます。

引数

path ファイルの保存場所。引用符付きで指定します。ファイル名には拡張子 .pptx を含める必要があります。同じ名前のファイルがある場合は上書きされます。

template（オプション）独自の PowerPoint テンプレートのパス名とファイル名。引用符付きで指定します。この引数が指定されていない場合は、インストールディレクトリ内の pptx フォルダにあるデフォルトのテンプレートが使用されます。

テンプレートには簡単なテーブルを含めるようにしてください。そうでない場合は、レポートテーブルにデフォルトのテーブル形式が適用されます。Windows での例については、JMP インストールフォルダの /pptx/JMPExportTemplate.pptx を参照してください。

Insert（オプション）既存のプレゼンテーション内でスライドを挿入する位置を指定する。

- *n* は、スライドを *n* 番目のスライドとして挿入することを意味します。
- "Begin" は、スライドをプレゼンテーションの冒頭に挿入します。
- "End" は、スライドをプレゼンテーションの末尾に挿入します。

Replace（オプション）既存のプレゼンテーション内で交換するスライドを指定する。引数は、Insert の場合と同様に、*n*、"Begin"、"End" です。

Append（オプション）スライドを既存のプレゼンテーションの末尾に挿入する。

Outline Titles（オプション）スライドのアウトラインタイトルの位置。デフォルトでは、アウトラインタイトルはスライドの左下隅に表示されます。

- "None" は、グラフィックの上のスライドタイトルとアウトラインタイトルを省略します。
- "Hide" は、アウトラインタイトルを省略します。
- "TopLeft"（左上）、"TopRight"（右上）、"BottomLeft"（左下）、"BottomRight"（右下）により、スライド上のアウトラインタイトルの位置が決まります。アウトラインタイトルとその親タイトルが含まれます。

format（オプション）埋め込みグラフィックの形式。引用符付きで指定します。オプションは、"Native"、"EMF"、"PNG"、"JPG"、"BMP"、"GIF"、"TIF" です。Windows の場合、ネイティブ形式は EMF です。Macintosh の場合、ネイティブ形式は PDF です。互換性の問題については、「ノート」を参照してください。この引数を指定しなかった場合は、［一般］環境設定の［PowerPoint のイメージ形式］での設定が適用されます。

ノート

Windows は、Macintosh で作成されたネイティブ PDF グラフィックをサポートしません。Macintosh は、Windows で作成されたネイティブ EMF グラフィックをサポートしません。クロスプラットフォームの互換性を保つためには、"PNG"、"JPG"、"GIF"、または "TIF" を指定してください。

```
db<<Save RTF("" | "pathname", "format")
```

RTF ソースファイルを保存する。指定された形式（"*format*"）のグラフィックファイルを格納したフォルダも一緒に保存されます。

```
db<<Save Text("" | "pathname", "format")
```

ディスプレイボックスのテキストを含んだファイルを保存する。

```
db<<Scroll Window(x, y)
```

ディスプレイボックスが表示されているウィンドウをスクロールする。

```
db<<Select
db<<Deselect
```

ディスプレイボックスを選択（強調表示）、または選択解除する。

```
db<<Set Menu Item State(index, 0 | 1 | -1)
```

指定されたポップアップメニュー項目の状態を設定する。通常（0）、選択されている（1）、選択不可（-1）のいずれか。

```
db<<Set Page Setup<margins(left, right, top, bottom)>,
<scale(s)>,<portrait(Boolean)>, <paper size("paper"))
db<<Set Page Setup<margins({left, right, top, bottom})>,
<scale(s)>,<portrait(Boolean)>, <paper size("paper"))
```

ページ設定を行う。margins（余白）はセンチ単位で指定します。scale 変数 *s* は、10（1000%）から 0.2（20%）の範囲の数値で、デフォルト値は 1（100%）です。portrait が真（1）の場合、ページは縦向き、それ以外の場合は横向きです。paper size には、用紙サイズ（"Letter"、"Legal" など）を指定します。

例

下の例では、新しいウィンドウを作成し、ページ設定の各項目を設定します。

```
w = New Window( "Window",
   Text Box( "Page Setup Test" )
);
w << Set page setup(
   margins( 1, 1, 1, 1 ),
```

```
    scale( 1 ),
    portrait( 1 ),
    paper size( "Letter" )
);
```

db<<Set Submenu (index, submenu count)

index で指定した番号のメニュー項目に対して、指定した数のサブメニュー項目を設定する。

例

下の例は、"A"、"B"、"C" というメニュー項目があるメニューを作成します。"A" にはサブメニュー項目 "A1" と A2" を、"B" にはサブメニュー項目 "B1"、"B2"、"B3" を設定しています。

```
New Window( "title", ob = Outline Box( "title" ) );
ob << Set Menu Script(
   {"A", "", "A1", Print( "A1" ), "A2", Print( "A2" ),
   "B", "", "B1", Print( "B1" ), "B2", Print( "B2" ), "B3", Print( "B3" ),
   "C", Print( "C" )}
);
ob << Set Submenu(1, 2); // メニュー A、サブメニューに A1 と A2 の 2 項目
ob << Set Submenu(4, 3); // メニュー B、サブメニューに B1、B2、B3 の 3 項目
```

db<<Set Report Title("title")

レポートの新しいタイトルを設定する。

Show Properties(db)

与えられたディスプレイボックスで利用できるメッセージを表示する。

db<<Sib

ボックスの兄弟を戻す。

db<<Sib Append(db2)

ディスプレイボックス db と同レベルにディスプレイボックス db2 を追加する。引数は、評価結果がディスプレイボックスへの参照である必要があります。

db<<Size Window(x, y)

ディスプレイボックスが表示されているウィンドウのサイズを変更する。

db<<Update Window

ディスプレイボックスを（オペレーティングシステムによっては、他のウィンドウも同時に）保持するウィンドウに無効化された領域がある場合に、そのウィンドウを更新する。無効化されたボックス領域には、新しいコンテンツが再描画されます。

ノート

スライダと JSL コールバックを組み合わせた一部のインタラクティブな JSL スクリプトでは、<<Update Window を使用して、ディスプレイの各部をスライダと同期させる必要があります。

db<<Zoom Window

内容がすべて表示されるようにウィンドウのサイズを拡大する。

Axis Box

axis box<<Axis Settings(<named_arguments>)

「軸の指定」ウィンドウを開くか、または、目盛りや軸ラベルなどの軸の設定を指定する。
引数が指定されていない場合は、「軸の指定」ウィンドウを表示します。
そうでない場合は、各軸に名前付き引数を指定します。

- Y軸は axis box(1) として指定します。
- X軸は axis box(2) として指定します。

次の例は、二変量に対する散布図を作成し、X 軸と Y 軸の設定を定義します。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
biv = dt << Bivariate( X( :Name("身長(インチ)") ), Y( :Name("体重(ポンド)") ),
   FitLine );
rbiv = biv << Report;
xaxis = rbiv[axis box( 2 )];
yaxis = rbiv[axis box( 1 )];
xaxis << Axis Settings( Show Major Grid( 1 ) );
yaxis << Axis Settings( Decimal( 10, 3 ) );
```

axis box<<Add Axis Label("string")

軸に、指定された文字列のラベルをつける。

axis box<<Add Ref Line(*number*, "*linestyle*", <"color">, <"label">, <width>)

指定の数値（*number*）の位置に、指定の線種（*linestyle*）("Solid" | "Dashed" | "Double")、色（"*color*"）("*color name*" または*インデックス番号*)、ラベル（"*label*"）、幅（*width*）（単位はピクセル）で参照線を引く。**メモ：**既存の参照線と同じラベル（"*label*"）の参照線を指定すると、既存の参照線が削除され、新しい参照線が引かれます。

axis box<<Decimal(*width*, *decimalplaces*)

軸の数値の表示形式を変更する。

axis box<<Format("name")

指定の名前（*"name"*）で与えられた表示形式に変更する。

axis box<<Get Inc(*n*)

軸の目盛りの間隔を取得する。

axis box<<Inc(*n*)

目盛りの間隔を設定する。

axis box<<Interval(*"format"*)

Inc()（目盛り間隔）で使われる値の単位を、日付／時間データの形式で指定する。"Numeric"、"Year"、"Quarter"、"Month"、"Week"、"Day"、"Hour"、"Minute"、または "Second"。

axis box<<Label Orientation(*"format"*)

説明

軸のラベルを、引用符付きで指定された向きに回転させる。指定できる引数は、"Automatic"（自動；ラベルの幅に基づく）、"Horizonal"（横）、"Vertical"（縦）、"Perpendicular"（垂直）、"Parallel"（平行）、"Angled"（角度）。

axis box<<Major Grid Line Color("*color*")

色の名前（"*color name*"）またはインデックスのどちらかを使って主目盛りのグリッド線（使用可能な場合）の色を設定する。

axis box<<Max(max)

軸に表示される最大値を変更する。

axis box<<Minor Grid Line Color("*color*")

色の名前（"*color name*"）またはインデックスのどちらかを使って補助目盛りのグリッド線（使用可能な場合）の色を設定する。

axis box<<Min(min)

軸に表示される最小値を変更する。

axis box<<Minor Ticks(*number*)

目盛りの間に刻む補助目盛りの数（*number*）を指定する。

axis box<<Remove Axis Label

Add Axis Label でつけられたラベルをすべて削除する。

axis box<<Reverse Scale(Boolean)

通常のスケールの方向を逆にして、最大値が左または下に来るようにする。

axis box<<Revert Axis

軸を元の設定（作成時の設定）に戻す。

axis box<<Scale("type")

軸のスケールを指定のタイプ（*type*）に変更する。選択肢は、"Linear" | "Log" | "Exp Prob"| "Weibull Prob" | "Logistic Prob" | "Frechet Prob" | "Normal" | "Cube Root" | "Johnson Su Scale" | "Geodesic" | "Geodesic US" | "Custom Scale" | "Power"。

タイプ（type）に Custom Scale を指定する場合、Scale to Internal(expr) と Scale to External(expr) の 2 つの節も指定する必要があります。

axis box<<Tick Font("name", <size>, <"style" | "style style...">, "<angle>")

目盛りのフォントとプロパティを設定する。複数のスタイルを指定するには、各スタイルの間にスペースを挿入し、スタイルを引用符付きで指定します。

axis box<<Show Labels(Boolean)

軸上の値のラベルを表示する、または表示しない。

axis box<<Show Major Grid(Boolean)

目盛りに合わせてグリッド線を引く、または削除する。

axis box<<Show Major Ticks(Boolean)

目盛りを表示する、または削除する。

axis box<<Show Minor Grid(Boolean)

補助目盛りに合わせてグリッド線を引く、または削除する。

axis box<<Show Minor Ticks(Boolean)

補助目盛りを表示する、または削除する。

axis box<<Tick Label List(<i>, {"text1","text2", ...},{n1, n2, ...})

軸の目盛りラベルの値と位置を設定する。

メモ: 目盛りの位置を指定しないと、間隔は、自動的に 1.0 に設定されます。

最初の整数 i（オプション）にはラベル行のインデックスを指定します。指定しない場合は、既存のラベル行がクリアされ、指定のとおりに新しく作成されます。指定すると、特定のラベル行が上書きされます。現在のラベル行数より大きいインデックスを使用すると、末尾に新しいラベル行が追加されます。

2 番目のリスト引数にはラベルの文字列タイトルを指定します。

3 番目のリスト引数はオプションで、各ラベルに対応する値を指定できます。値のリストを省略した場合、ラベルは 1 から 1 つずつ増える整数となります。

Border Box

メモ: Boader Box でサポートされているディスプレイボックスの引数は、1つだけです。

border box<<Set Background Color({R, G, B} | <"color">)

境界ボックスの背景色を設定する。色名（"*color name*"）または RGB 値のリスト値を指定します。たとえば、次のように実行します。

```
border box<<set background color("red");
```

または

```
border box<<set background color( {255, 192, 3} );
```

border box<<Set Color({R, G, B} | <"color">)

境界ボックスの境界線の色を設定する。色名（"*color name*"）または RGB 値のリスト値を指定します。たとえば、次のように実行します。

```
border box<<set color("red");
```

border box<<Get Color

境界ボックスの境界線の色を取得する。

border box<<Set Style ("style")

境界ボックスの境界線の線種を設定する。線種は、0 ／ "Solid"（実線）、1 ／ "Dotted"（点線）、2 ／ "Dashed"（破線）、3 ／ "DashDot"（一点鎖線）、または 4 ／ "DashDotDot"（二点鎖線）のいずれかで指定します。たとえば、次のように実行します。

```
border box<<Set Style("dotted");
```

border box<<Get Style

境界ボックスの境界線の線種を取得する。たとえば、次のように実行します。

```
border box<<Get Style;
```

Data Grid Box

dgb<<Set Data Table(<data_table>)

データグリッドボックスのデータテーブルを設定する。

Frame Box

frame box<<Add Graphics Script(<order>,<"description">, <script>)

説明

フレームボックス内にグラフィックを描くためのスクリプトを追加する。

引数

order（オプション）グラフィック要素を描く順序を指定する。指定できる値は、Back または Forward、あるいは複数のグラフィック要素の描画順を示す整数です。1は、そのオブジェクトを最初に描くことを示します。

description（オプション）「グラフをカスタマイズ」ウィンドウのグラフィックスクリプトの横に表示される文字列。

script（オプション）JSL スクリプト。

例

次の例では、グラフィックスクリプトはまず線を描き、次に二変量の散布図に必要なその他のグラフィック要素（グリッド線、参照線、マーカー）を描きます。1の順序引数を指定しなかった場合は、線が最後に描かれてマーカーを隠してしまいます。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
obj = dt << Bivariate( Y( :Name("体重(ポンド)") ), X( :Name("身長(インチ)") ) );
Report( obj )[FrameBox( 1 )] <<
Add Graphics Script(
   1, // 最初に線を描く
   Description( "Pen Script" ),
   Pen Color( "red" );
   Pen Size( 5 );
   Y Function( 60 + 120 / 2 * (1 + Sine( (2 * Pi() * (x - 50)) / 22.5 )), x );
);
```

frame box<<Append Seg

特定のフレームボックスにディスプレイセグメントを追加する。

frame box<<Background Color({R, G, B} | <"color">)

背景の色を変更する。色名（"*color name*"）またはRGB値のリスト値を指定します。

frame box<<Child Seg

フレームボックスのディスプレイセグメントの子を戻す。

frame box<<Edit Graphics Script

現在のグラフィックスクリプトの編集や削除を行うためのダイアログボックスを表示する。

frame box<<Find Seg

指定の引数を持つディスプレイセグメントを戻す（たとえば、セグメントの名前）。

frame box<<Frame Size(x, y)

ピクセル値をもとに、フレームのサイズを変更する。

frame box<<Make table of graphs like this

グラフを含むデータテーブルを作成する。

frame box<<Marker Size(size)

マーカーのサイズを変更する。

frame box<<Row Colors(color)

frame box<<Row Markers(marker)

frame box<<Row Exclude

frame box<<Row Hide

frame box<<Row Label

レポートに関連するデータテーブルにコマンドを送る。これらのメッセージにより、選択された行の属性が変更されます。<<Row Exclude、<<Row Hide、および <<Row Label では、引数を省略すると、オプションが切り替わります。オプションがオフの場合はオンになり、オンの場合はオフになります。

frame box<<X Axis(min, max, inc, named arguments)

X軸のスケールを設定する。

frame box<<Y Axis(min, max, inc, named arguments)

Y軸のスケールを設定する。

Matrix Box

matrix box<<Get

行列の要素を戻す。

matrix box<<Make Into Data Table(<Invisible(Boolean)>)

行列をもとに、新しいデータテーブルを作成する。オプションの引数 Invisible() が True の場合、データテーブルは非表示になります。False の場合、データテーブルは表示されます。

matrix box<<Set Format(width, decimalplaces)

行列の要素の数値形式を指定する。

matrix box<<Sort(column_num, ascending_bool)

column_num で指定された列番号に基づいて行列の行を並べ替える。デフォルトの順序は昇順です。

column_num が 0 の場合、その並べ替えは削除されます。

ascending_bool が "True" の場合は昇順、"False" の場合は降順に並び替えられます。

Nom Axis Box

nom axis box<<Divider Lines(Boolean)

軸ボックスのラベルを区切る分割線を引く、または消す。

nom axis box<<Lower Frame(Boolean)

軸周辺の下のフレームを追加または削除する。

nom axis box<<Rotated Tick Labels(Boolean)

各目盛りの値のラベルを回転させる、または回転させない。

Number Col Box

number col box<<Add Element(item)

項目（item）を数値列ボックスに追加する。項目は 1 つの数字、数字のリスト、または数字の行列です。

number col box<<Get

number col box<<Get(i)

値すべてをリストで取得する。または、*i*番目の値を取得する。

number col box<<Get As Matrix

値を行列（列ベクトル）の形で取得する。

number col box<<Get Format

現在の表示形式（桁数（*width*）、小数点以下桁数（*decimalplaces*））を戻す。小数点以下桁数（*decimalplaces*）が 100 より大きいときは、時間／日付の値であることを示します。

number col box<<Get Heading

列のタイトルのテキストを戻す。

number col box<<Set Format(width, decimalplaces)

number col box<<Set Format("format", width)

表示形式を設定する。小数点以下桁数（*decimal places*）が 100 より大きいときは、時間／日付の値であることを示します。表示形式を *p* 値形式にする場合は、*decimalplaces* に 97 を指定します。

たとえば、次のようになります。

```
<< Set Format( 10, 2)
<< Set Format( "Scientific", 10 )
```

1000 以上の値にカンマを追加するには、"Use thousands separator" という引数を追加します。

number col box<<Set Heading("string")

列のタイトルのテキスト（text）を変更する。

Outline Box

outline box<<Close(Boolean)

アウトラインボックスを閉じる。

outline box<<Close All Below

ノードの子ノードをすべて閉じる。

outline box<<Close All Like This

このアウトラインボックスと同様のノードをすべて閉じる。

outline box<<Close Where No Outlines

子ノードを持たないノードをすべて閉じる。

outline box<<Get Title

アウトラインボックスのタイトルを取得する。

outline box<<Horizontal(Boolean)

ノードの子ノードを横に並べる。

outline box<<Open All Below

ノードの子ノードをすべて開く。

outline box<<Open All Like This

このアウトラインボックスと同様のノードをすべて開く。

outline box<<Set Menu Script({"string1", script1,"string2", script2, ...})

赤い三角ボタンをクリックしたときのメニューに項目を追加する。

outline box<<Set Title("text")

アウトラインボックスのタイトルを変更する。

Panel Box

panel box<<Get Title

パネルボックスのタイトルを取得する。

panel box<<Set Title("text")

パネルボックスのタイトルを変更する。

Plot Col Box

`plot col box<<Get As Matrix`

値を行列（列ベクトル）の形で取得する。

`plot col box<<Set Values([matrix] or {list})`

行列（数値変数の場合）またはリスト（文字変数の場合）の値を列の値として設定する。

Slider Box および Range Slider Box

`slider box<<Get(<index>)`

`range slider box<<Get Lower(<index>)`

`range slider box<<Get Upper(<index>)`

スライダの現在の値を戻す。

`slider box<<Set(n, <index>, <run script(0|1)>)`

`range slider box<<Set Lower(n, <index>, <run script(0|1)>)`

`range slider box<<Set Upper(n, <index>, <run script(0|1)>)`

スライダの値を設定する。`run script(0|1)` は、`<<Set`、`<<Set Lower`、または `<<Set Upper` メッセージの後に on-change スクリプトを実行するかどうかを制御します。

`slider box<<Get Min()`

範囲スライダとスライダに使用できる最小値を戻す。

`slider box<<Set Min(float, <index>)`

範囲スライダとスライダに使用できる最小値を設定する。

`slider box<<Get Max()`

範囲スライダとスライダに使用できる最大値を戻す。

`slider box<<Get Var`

`range slider box<<Get Lower Var`

`range slider box<<Get Upper Var`

スライダの値が入る変数の名前を戻す。

slider box<<Set Max(float, <index>)

範囲スライダとスライダに使用できる最大値を設定する。

slider box<<Set Script(<script>)

範囲スライダとスライダの更新時に実行するスクリプトを設定する。

slider box<<Set Var(slider_var)

range slider box<<Set Lower Var(slider_var)

range slider box<<Set Upper Var(slider_var)

スライダの値が入る変数の名前を設定する。

String Col Box

string col box<<Add Element(item)

項目（item）を文字列ボックスに追加する。項目は1つの引用符付き文字列、引用符付き文字列のリスト、または引用符付き文字列の行列です。

string col box<<Get

string col box<<Get(i)

リスト内の値すべて、または *i* 番目の値を取得する。

string col box<<Get Heading

列のタイトルのテキストを戻す。

string col box<<Set Heading("text")

列のタイトルのテキスト（*text*）を変更する。

string col box<<Set Justify("right"|"left" |"center")

文字列ボックス内の文字列の位置を右寄せ（right）、左寄せ（left）、中央寄せ（center）のいずれかに指定する。

Tab Box

tab box<<Get Tab Margin()

タブボックスの現在のマージン（単位はピクセル）をリストで戻す。戻り値のリストは、{左，上，右，下}の順番で、現在のマージンを含んでいます。

tab box<<Set Style("tab" | "combo" | "outline" | "vertical spread" | "horizontal spread" | "minimize size")

タブボックスの表示形式をタブからコンボボックスまたはアウトラインノードに変更する。

"vertical spread" と "horizontal spread" は、タブタイトルの表示の向きを変更します。

"minimize size" はタブのスタイルがタイトルの幅に応じて決まります。例については、『スクリプトガイド』の「表示ツリー」章を参照してください。

tab box<<Set Tab Margin(n|{...})

タブボックスのタブのマージンを設定する。数字が1つだけ指定された場合は、4つのマージンすべてをこのピクセル数に設定します。2つの数字のリストが指定された場合は、左右のマージンを最初の数字、上下のマージンを2番目の数字のピクセル数に設定します。4つの数字のリストが指定された場合は、マージンを{左，上，右，下}の順序で設定します。

tab box<<Show Tabs(0|1)

タブボックスのタブの表示／非表示を切り替える。タブを非表示にした場合、タブを選択したり表示したりする別の方法が必要になります。たとえば、タブへの参照リストを含んだリストボックスなどです。デフォルト値は1です。

Table Box

table box<<Get

表に含まれている値をリストの形で取得する。

table box<<Get As Matrix

表に含まれている数値を行列の形で取得する。

table box<<Get Locked Columns

手のひらカーソルでドラッグできない列、または前に列をドロップできない列の数を戻す。

table box<<Get Row Change Function

行が選択されたときに評価する式を戻す。

table box<<Get Selectable Rows

テーブルボックスの行が現在、選択可能な場合は真（1）を戻す。

table box<<Get Selected Row Color

テーブルボックス内の選択された行の背景色のインデックス番号を戻す。

table box<<Make Combined Data Table

Make Data Table と同じ。ただし、同じ列を持つレポートテーブルを検索し、これらをすべて組み合わせて新しいデータテーブルを作成します。

table box<<Make Data Table(name)

表の内容を新しいデータテーブルに出力する。

table box<<Set Row Change Function

行が選択されたときに評価する式を設定する。

table box<<Set Cell Changed Function(Function({this, col box, row},<script>;))

テーブル内の列のセルをユーザーが編集したときに必ず呼び出される関数を設定する。

例

この例では、変更されたセルの変更後の値をログに出力します。

```
New Window( "Mountains",
   tb = Table Box(
      String Col Edit Box(
         "Mountain",
         {"K2", "Delphi", "Kilimanjaro",
         "Grand Teton"}
      ),
      Number Col Edit Box(
         "Elevation (meters)",
         {8611, 681, 5895, 4199}
      ),
      Plot Col Box( "", {8611, 681, 5895, 4199} )
   )
);
tb <<
Set Cell Changed Function(
   Function( {this, col, row},
      Print(
         (col << Get Heading) || ": row:" ||
         Char( 3 ) || " is now " ||
```

```
            Char( col << Get( row ) )
        )
    )
);
```

table box<<Set Locked Columns(n)

最初の n 列をロックする。ロックされた列は、手のひらカーソルでドラッグしたり、前に列をドロップしたりできません。

table box<<Set Selectable Rows(Boolean)

テーブルボックスの行を選択可能または選択不可能にする。

table box<<Set Selected Row Color("color")

テーブルボックスの行が選択可能な場合（Set Selectable Rows(True)）、選択された行の背景色を設定する。

Text Box

text box<<Font Color(n)

テキストの色を指定する。

text box<<Get Hidden State

テキストボックスの現在の状態を戻す。

text box<<Get Text

ボックス内のテキストの内容を戻す。

text box<<Get Tip

テキストボックス（またはテキスト編集ボックス）のツールヒントを戻す。

text box<<Markup

指定した HTML タグによりフォーマットされたテキストを戻す。HTML は完全な形でなければなりません。入れ子状のタグは適切に閉じてください。

次の例では、太字、下線、斜体のテキストが戻されます。

```
w = New Window( " 書式付きテキスト ",
   Text Box( " これは <b> 太字 </b> のテキストです。これは <b><i> 太字斜体 </i></b> のテキストです。これは <u> 下線付き </u> のテキストです。",
   <<Markup) );
```

text box<<Rotate Text("direction")

テキストを左（"left"）または右（"right"）に 90 度回転するか、水平に戻す。

text box<<Set Font("name", <size>, <style | "style style...">, <angle>)

テキスト文字列のフォント名と属性を設定する。複数のスタイルを指定するには、各スタイルの間にスペースを挿入し、スタイルを引用符付きで指定します。

text box<<Set Font Size(n)

テキストのフォントサイズをポイント数で設定する。

text box<<Set Font Style("style" | "style style...")

テキストのフォントスタイルを設定する。複数のスタイルを指定するには、各スタイルの間にスペースを挿入し、スタイルを引用符付きで指定します。互換性のためスタイルを整数で指定することも可能ですが、新しいスクリプトにはお勧めしません。

text box<<Set Script(script)

スクリプトをテキストボックスに関連付ける。ユーザが Enter キーを押すと（または、テキスト編集ボックスがフォーカスを失ったときに）、スクリプトが実行されます。

text box<<Set Text("string")

ボックス内の文字列を変更する。

text box<<Set Tip("string")

テキストボックス（またはテキスト編集ボックス）のツールヒントを設定する。

text box<<Set Wrap(n)

改行ポイントをピクセル（*n*）で設定する。

ツリーノードとツリーボックス

以下のメッセージで、node はツリーノードまたはツリーノードへの参照、root はツリーボックスまたはツリーボックスへの参照を表します。

node <<Append(<node_ref>)

このノードの子の後ろにツリーノードを挿入する。

root <<Collapse(<node>)

ツリー内のノードを折りたたむ。

root <<Expand(<node>)

ツリー内のノードを展開する。

root <<Get Selected(<node>)

現在選択されているツリーノードを取得する。

- 項目が1つのツリーでは、現在選択されているツリーノードまたはEmptyが戻されます。
- MultiSelect引数を含む場合のTree Box()の結果は、表3.1に示すとおりです。

表3.1 Multi-Selectを使った結果

ツリー内の選択項目数	結果
選択項目なし	empty
1つの項目を選択	1つのツリーノードのリスト
複数の項目を選択	選択された複数のツリーノードのリスト

node <<Get Tip

ツリーノードのツールヒントを戻す。

root <<Is Multiselect

複数選択ツリーの場合は1、単一選択のツリーの場合は0を戻す。

node <<Prepend(<node_ref>)

このノードの子の前に、ツリーノードを挿入する。

node <<Remove(<node>)

ツリーから指定されたノードとそのすべての子を削除する。

root <<Set Selected(node|{nodes}, <0|1>)

指定されたツリーノードを、ツリーディスプレイボックス内で選択する。ツリーノードのリストが指定された場合、複数選択のツリーではリスト内のすべてのノードが選択されます。そうでない場合は、リストの最初のノードが選択されます。ノードの選択、または非選択は、ブール引数で指定します。デフォルト値は1で、ノードが選択されます。

メモ:

- Windowsの場合、<<Set Selected()メッセージは、選択されたノードとツリーのルートとの間のすべてのノードを展開し、ツリーの奥深くで選択されている項目をすべて表示します。この展開の状態は、すでに選択されていたノードには影響しません。
- Macintoshの場合、<<Set Selected()はツリーの展開の状態を変更しません。

`node <<Set Tip("text")`

ツリーノードのツールヒントを設定する。

三角分割

以下のメッセージで、triは三角分割または三角要素座標への参照を表します。

`tri <<Get N Points`

三角分割において、一意な点の個数を戻す。

`tri <<Get Points`

三角分割において、一意な点の座標を戻す。

`tri <<Get Y`

三角分割において、一意な各座標に対するY値を戻す。

`tri <<Get N Hull Points`

三角分割において、凸包を構成する点の個数を戻す。

`tri <<Get Hull Points`

三角分割の境界の点のインデックスを戻す。

`tri <<Get N Hull Edges`

三角分割において、凸包を構成する辺の数を戻す。

`tri <<Get Hull Edges`

三角分割において、凸包を構成する辺の通し番号を戻す。

`tri <<Get Hull Path`

三角分割において、凸包の情報をパス形式で戻す。

tri <<Get N Triangles

三角形の個数を戻す。

tri <<Get Triangles

三角分割において、各三角形の情報を Nx3 の行列で戻す。

tri <<Get N Edges

三角分割の辺の数を戻す。

tri <<Get Edges

三角分割において、辺のインデックスを Nx2 の行列で戻す。

tri <<Subset({indices})

指定された点の部分集合に対して、その三角分割を戻す。

tri <<Peel

現在の三角分割から、凸包を構成する辺（一番外側の辺）を削除する。

Window

window<<Bring Window to Front

ウィンドウを最前面に移動する。

window<<Close Window(<nosave>)

ウィンドウを閉じる。オプションの引数 nosave を指定した場合は、保存したり、確認のメッセージが表示されたりしないまま、ウィンドウ（ジャーナルやレポートなど）が閉じます。

window<<Get Content Size

ウィンドウの内容のサイズを戻す。

window<<Get Window Icon

ウィンドウのアイコンの名前を戻す。

window<<Get Window Position

ウィンドウの位置を戻す。

window<<Get Window Size

ウィンドウのサイズを戻す。

window<<Get Window Title

ウィンドウのタイトルを戻す。

window<<Inval

ディスプレイボックスを無効にする。ウィンドウは、<<Update Window メッセージが送られた際、または、次回、オペレーティングシステムによってウィンドウが更新される際に更新されます。その他の方法については、<<Reshow を参照してください。

window<<Maximize Display

ウィンドウを最大化する。

window<<Maximize Window(Boolean)

ウィンドウを最大化する。

window<<Minimize Window(Boolean)

ウィンドウを最小化する。

window<<Move Window(x, y)

ウィンドウを指定の位置に移動させる。

window<<On Close(script)

ウィンドウが閉じられたときにスクリプトを実行する。

window<<Pad Window(Boolean)

ウィンドウの内容に合わせてパディング（余白）をオン（1）またはオフ（0）にする。デフォルト値はオフです。

window<<Print Window

ウィンドウをデフォルトのプリンタに印刷する。「印刷」ウィンドウは開かれず、ユーザによる入力が不要です。

window<<Reshow

ディスプレイボックスを無効にして、ウィンドウを新しいコンテンツで更新する。更新のタイミングをより具体的に制御する必要がある場合は、<<Inval メッセージおよび <<Update Window メッセージを参照してください。

window<<Set Main Window

指定のウィンドウを、JMP の実行時に表示されるデフォルトのウィンドウとして設定する。

window<<Set Window Icon("icon name")

ウィンドウのアイコンを指定のとおりに設定する。

window<<Set Window Size(x, y)

ウィンドウのサイズを変更する。

window<<Show Window(0|1)

1 はウィンドウを表示し（ウィンドウが現在開かれていない場合）、0 はウィンドウを非表示にする。ウィンドウが（Windows で）最小化されているか（Macintosh で）固定されている場合は、ウィンドウを表示すると通常の状態に戻され、最前面に表示されます。

window<<Size Window(x, y)

ウィンドウのサイズを変更する。

window<<Update Window

ディスプレイボックスに無効になった領域がある場合、そのディスプレイボックスを保持するウィンドウを更新または再描画する。その他の方法については、<<Inval メッセージおよび <<Reshow メッセージを参照してください。

window<<Window Class Name

ディスプレイボックスのウィンドウのクラス名を戻す。有効な応答は、DataTable、FormulaEditor、Starter、Journal、Layout、Launcher、Report、Dialog、DialogWithMenu、ModalDialog、FindReplace、User、Generic、ToolWindow、FindReplace、AppBuilder、Debugger です。

window<<Zoom Window

内容がすべて表示されるようにウィンドウのサイズを拡大する。

ダイナミックリンクライブラリ（DLL）

dll object<<Call DLL(function_name, signature, arguments)

指定のシグニチャおよび引数で、DLL 内の指定の関数を呼び出す。

dll object<<Declare Function("name", <named_arguments>)

指定の関数における戻り値と引数の型を宣言する。この宣言により、DLL 内の関数を正しく呼び出すことができます。名前付き引数 Convention には、STDCALL、PASCAL、CDECL のいずれか 1 つを指定します。また、Return および Arg において、戻り値と引数の型を指定します。

名前付き引数

Alias("name") DLL にエンコードされた名前を使用したくない場合、別名を指定することができる。

Arg(type, <"description">, <access_mode>, <array>) Arg（オプション）は、関数がもつ引数ごとに 1 回ずつ、複数回使用できます。

type には、次のいずれかのキーワードによって、引数の型を指定します。Int8、UInt8、Int16、UInt16、Int32、UInt32、Int64、UInt64、Float、Double、AnsiString、UnicodeString、Struct、IntPtr、UIntPtr、ObjPtr

"description"（オプション）は、引数に対する何らかのコメントを記載するための文字列です。

access_mode（オプション）は、引数の渡し方を指定するキーワードです。input は、値渡しを意味します。引数の値を渡す時に指定します。output は、アドレス渡しを意味します。値が定義されていない引数のアドレスを渡す時に指定します。update は、参照渡しを意味します。引数への参照を渡す時に指定します (この時の引数の値は、JSL 変数の値となります)。デフォルト値は input です。

array はオプションのキーワードです。このオプションは、type が Double で、access_mode が input または update に指定されている場合のみ有効です。関数の引数が、Double の配列であることを指定します。

Convention(calling_convention) 呼び出しの規則を指定するオプションのキーワード。STDCALL、PASCAL、または CDECL のいずれかを指定します。デフォルト値は STDCALL です。STDCALL と PASCAL は等価です。

MaxArgs(n)（オプション）使用可能な引数の最大数を整数で指定する。

MinArgs(n)（オプション）使用可能な引数の最小数を整数で指定する。

Returns(type)（オプション）関数からの戻り値の型を指定する。Int8、UInt8、Int16、UInt16、Int32、UInt32、Int64、UInt64、Float、Double、AnsiString、UnicodeString、Struct、IntPtr、UIntPtr、ObjPtr のいずれか。

StackOrder(order)（オプション）関数の呼び出し時における、スタックへの引数の格納順序を指定するキーワード。有効な値は L2R（左から右）、R2L（右から左）です。デフォルト値は R2L です。

StackPop(pop)（オプション）エクスポートされた関数が戻った後にスタックをクリアする際の方法を指定するキーワード。有効な値は CALLER および CALLEE です。デフォルト値は CALLEE です。

StructArg(Arg(...), <Arg(...)>, ..., <access_mode>, <pack_mode>, <"description">)（オプション）複数回使用できる。エクスポートした DLL 関数において、構造体の引数を引数として渡す必要がある場合は、StructArg を使用して構造体メンバーを宣言します。

Arg 引数の構文は、前述の DeclareFunction における Arg 引数と同じです。他の引数として、access_mode および pack_mode があります。

access_mode（オプション）は、構造体引数が値（input）によって渡されるか、または参照（update）によって渡されるかを指定するキーワードです。

pack_mode（オプション）は、構造体のパック方法を指定するための整数です。有効な値は 1、2、4、8、16 です。デフォルト値は 8 です。

"description"（オプション）は、構造体に対する何らかのコメントを記載するための引用符付き文字列です。

dll object<<Get Declaration JSL

DLL 内に用意された JSL による関数宣言をログに出力する。

dll object<<Load DLL("path" <,AutoDeclare(Boolean | Quiet | Verbose)>)

dll object<<Load DLL("path" <, Quiet | Verbose>)

path で指定された DLL をロードする。

path DLL をロードする場所のパス名。引用符付きで指定します。

AutoDeclare(Boolean | Quiet | Verbose)（オプション）AutoDeclare(1) および AutoDeclare(Verbose) は、ログに verbose メッセージを書き込む。AutoDeclare(Quiet) は、ログウィンドウのメッセージを無効にします。このオプションを省略すると、ログに verbose メッセージが書き込まれます。

Quiet | Verbose（オプション）Declare Function() を使用した場合、このオプションは、ログウィンドウへのメッセージ出力のオフ（Quiet）とオン（Verbose）を切り替える。

dll object<<Show Functions

DLL オブジェクトに対して宣言された関数をログに送る。

dll object<<Unload DLL

DLL をアンロードする。

イメージ

イメージ処理の例については、［スクリプトの索引］を参照してください。JMP のメニューで、［**ヘルプ**］>［**スクリプトの索引**］を選ぶと、この対話的なヘルプを参照できます。

その他のリソースは、JMP File Exchange（https://community.jmp.com/community/file-exchange）で入手できます。

img<<Crop(Left(pix), Right(pix), Top(pix), Bottom(pix))

既存のイメージから、指定された大きさ（単位はピクセル）の新しいイメージを作成する。

img<<Filter("name", <n>)

指定されたアルゴリズムに基づいてイメージをフィルタリングする。フィルタリングはイメージ内のノイズを除去するのに便利です。

メモ: JMPのイメージフィルタはすべて、オペレーティングシステムレベルでサポートされます。Windowsで処理されたイメージとMacintoshで処理されたイメージは異なる場合があります。

引数

name JMPイメージフィルタの名前を示す引用符付き文字列。使用可能なフィルタは次のとおりです。

- "despeckle"は、スキャンまたはキャプチャされたイメージからしみ（引っかき傷やごみなど）を取り除く。
- "edge"は、明度が極端に変化する部分のピクセルを特定し、それらを暗くして極端な変化を抑える。エッジの検出は、表面、深さ、素材、およびライティングにおける変化の検出に使用されます。
- "enhance"は、ノイズが多いイメージでピクセル間のコントラストを低減する。
- "median"は、各ピクセルの明度を近接のピクセルと比較することで、ノイズ（ランダムな変動）を低減する。値が大きく異なるときは、近接ピクセルの平均値で置き換えます。
- "negate"は、各ピクセルの色をそれぞれ補色に変えることで、逆の色またはグレースケールのイメージを作成する。
- "normalize"は、カラーイメージのピクセルを、ファイル形式の数値システムの範囲全体を使用するよう変更する。この処理によって、イメージの色がより強調されます。
- "sharpen"は、ピクセルのエッジを目立たせることで、ぼけを低減する。
- "contrast", nは、イメージを明るくまたは暗くする。0.0より大きい値を指定するとイメージが明るくなり、0.0より小さい値を指定すると暗くなります。
- "gamma", nは、イメージの視覚的な表示（明度と彩度）を、モニターの違いを考慮して補正する。1.0より大きい値を指定するとイメージが明るくなり、1.0より小さい値を指定すると暗くなります。
- "reduce noise", nは、ISO感度が大きいときや露出時間が長いときに発生するイメージ内のランダムな変動（ノイズ）を低減する。
- "gaussian blur", radius, sigmaは、イメージのノイズやディテールを低減し、よりスムーズなイメージにする。半径（radius）は各ピクセルの周囲のぼかし半径で、sigmaはGauss分布の標準偏差です。このGaussぼかしは、通常、サイズ変更やエッジ検出の実行の際に使用されます。

img<<Flip Both

イメージの上下左右を反転する。

img<<Flip Horizontal

イメージの左右を反転する。

img<<Flip Vertical

イメージの上下を反転する。

img<<Get N Frames

マルチフレーム TIFF ファイルまたはアニメーション GIF ファイルに含まれるフレーム数を返す。フレーム番号は、0 から開始する。

例

次の例では、4 つのフレームからなる TIFF ファイルを新しいウィンドウに配置し、フレーム 1 のイメージを表示する。

```
img = New Image( "$DOWNLOADS/Multiframe.tif" );
nframes = img << Get N Frames(); // 4 を戻す
img << Set Current Frame( 1 ) // イメージ 1 を表示する
win = New Window( "Multi-Frame TIFF", img );
```

img<<Get Size

img<<Size

イメージの幅と高さ（単位はピクセル）を含んだリストを戻す。

img<<Rotate(degrees)

指定した角度だけイメージを回転させる。

img<<Save Image("filepath")

指定したパスとファイル名（filepath）でイメージを保存する。

img<<Scale(n, <n>)

イメージのサイズを指定の倍率で変更する。引数を 1 つ指定した場合は、幅と高さの両方が同じ倍率でサイズ変更されます。引数を 2 つ指定した場合は、幅と高さがそれぞれの倍率でサイズ変更されます。

例

```
img = New Image( "$SAMPLE_IMAGES/tile.jpg" );
xs = 2;
img << Scale( xs );
New Window( "Tilex 2", img );

img = New Image( "$SAMPLE_IMAGES/tile.jpg" );
img << Scale( 2, 0.5 ); // イメージの幅を 2 倍、高さを 1/2 倍にする
```

```
New Window( "Tile squished", img );
```

ノート

イメージのサイズを取得し、倍率を掛け、サイズを設定するスクリプトの代わりに Scale を使用できます。

img<<Set Current Frame

マルチフレーム TIFF ファイルまたはアニメーション GIF ファイルにおいて、表示したいフレーム番号を設定する。最初のフレーム番号は 0、最後のフレーム番号はフレーム数から 1 を引いた値となります。たとえば、フレームが 4 つある場合は、フレーム 0 ～フレーム 3 を指定できます。例については、「img<<Get N Frames」（323 ページ）を参照してください。

img<<Set Size(width, height)

イメージのサイズを指定の大きさ（単位はピクセル）に変更する。イメージの縦横比を固定したい場合は、元のイメージの縦横比と同じ比率の高さと幅を指定します。

img<<Transparency(fraction)

イメージの透明度を設定する。fraction には、0.0（完全に透明）～ 1.0（透明度なし）の値を指定します。

MATLAB

MATLAB 接続オブジェクトを使用して、MATLAB とのインターフェースをスクリプトで記述できます。MATLAB Connect() という JSL 関数を使用して、スクリプト可能な MATLAB 接続オブジェクトを取得できます。詳細については、『スクリプトガイド』の「JMP の拡張」章を参照してください。

mlconn << Control(<named arguments>)

MATLAB の実行を制御する。

戻り値

なし

名前付き引数

Echo(Boolean) グローバル。実行した MATLAB のプログラムコードを、JMP の「ログ」ウィンドウに出力する。

Visible(Boolean) グローバル。アクティブな MATLAB ワークスペースの表示／非表示を指定する。

mlconn << Disconnect()

説明

MATLAB 接続インターフェースを接続解除する。

mlconn << Execute({ list of inputs }, { list of outputs }, mCode, <named arguments>)

MATLABに対して、第1引数のリストに指定されたJMP変数を送り、第3引数に指定されたMATLABコードをサブミットする。第2引数のリストに指定されたMATLAB変数が、JMPに戻されます。

戻り値

成功した場合は0、そうでない場合は0以外。

引数

`{ list of inputs }` 位置固定、名前のリスト。入力としてMATLABに送られるJMP変数名のリスト。

`{ list of outputs }` 位置固定、名前のリスト。出力としてMATLABから取得されるJMP変数名のリスト。

`mCode` 位置固定、文字列。サブミットされるMATLABコード。

名前付き引数

`Expand(Boolean)` MATLABコードのサブミット前に、コードに対してEval Insertを実行する。

`Echo(Boolean)` 実行したMATLABのプログラムコードを、JMPの「ログ」ウィンドウに出力する。デフォルト値は1（真）です。

mlconn << Get Graphics(format)

MATLABのグラフウィンドウに最後に出力されたグラフを、指定の形式で取得する。format引数で指定されたグラフィック形式で取得する。

戻り値

JMPピクチャーオブジェクト。

引数

`format` 位置固定。MATLABのグラフを取得するときの変換形式。有効な形式は、`"png"`、`"bmp"`、`"jpeg"`、`"jpg"`、`"tiff"`、および`"tif"`です。

mlconn << Get Version()

インストールされているMATLABの現在のバージョンを取得する。

戻り値

行列。MATLABのバージョン番号を示す長さ3のベクトル。

引数

なし

mlconn << Get(name)

説明

name引数で指定されたMATLABの変数を、JMPで取得する。

戻り値
name 引数で指定された変数の値。

引数
name 位置固定。MATLAB から取得する JMP 変数の名前。

---

mlconn << Is Connected()

説明
接続がアクティブ化どうかを判断する。

戻り値
アクティブな MATLAB 接続がある場合は 1、そうでない場合は 0 を戻す。

引数
なし

---

mlconn << JMP Name To MATLAB Name(JMP name)

MATLAB の命名規則に従い、JMP 変数名を、対応する MATLAB 変数名に変換する。

戻り値
文字列。マップされた MATLAB 名。

引数
name 位置固定。MATLAB に送る JMP 変数の名前。

---

mlconn << Send(name, <named arguments>)

説明
名前付き変数を JMP から MATLAB に送る。

戻り値
成功した場合は 0、そうでない場合は 0 以外。

引数
name 位置固定。MATLAB に送る JMP 変数の名前。

名前付き引数
次の引数はデータテーブルにのみ指定可能です。

Selected(Boolean) 参照先データテーブルの選択されている行を MATLAB に送る。

Excluded(Boolean) 参照先データテーブルの除外されている行のみを MATLAB に送る。

Labeled(Boolean) 参照先データテーブルのラベルのついている行のみを MATLAB に送る。

Hidden(Boolean) 参照先データテーブルの非表示の行のみを MATLAB に送る。

Colored(Boolean) 参照先データテーブルの色のついている行のみを MATLAB に送る。

Markered(Boolean) 参照先データテーブルのマーカーのついている行のみを MATLAB に送る。

Row States (Boolean, <named arguments>) 参照先データテーブルにおける行属性の情報を、「RowState」という列名のデータ列を追加して、MATLAB に送る。個々の設定をあわせて追加すれば、複数選択も可能です。行属性の各設定には、それぞれ次の値が割り当てられています。

- Selected = 1
- Excluded = 2
- Labeled = 4
- Hidden = 8
- Colored = 16
- Markered = 32

Row States には、次の名前付き引数をオプションで指定できます。

`Colors(Boolean)` 行の色を送る。「RowStateColor」という名前のデータ列を追加します。

`Markers(Boolean)` 行のマーカーを送る。「RowStateMarker」という名前のデータ列を追加します。

mlconn << Submit(mCode, <named arguments>)

アクティブなグローバル MATLAB 接続に、MATLAB コードをサブミットする。

戻り値

成功した場合は 0、そうでない場合は 0 以外。

引数

`mCode` 位置固定、文字列。サブミットされる MATLAB コード。

名前付き引数

`Expand(Boolean)` MATLAB コードのサブミット前に、コードに対して Eval Insert を実行する。

`Echo(Boolean)` 実行した MATLAB のプログラムコードを、JMP のログに出力する。デフォルト値は 1（真）です。

mlconn << Submit File(pathname)

pathname で指定されたファイルに含まれる MATLAB コードを、MATLAB 上で実行する。

戻り値

成功した場合は 0、そうでない場合は 0 以外。

引数

`pathname` 位置固定、文字列。実行する MATLAB プログラムコードを含むファイルのパス名。

プラットフォーム

obj<<Action

式を評価する。複数のプラットフォームについて、それぞれ起動ウィンドウでユーザーの入力を受け取ってから順に実行するときに便利です。

obj<<Automatic Recalc

データの除外や変更があった場合、自動的に分析を再実行する。自動再計算がオンの場合、wait(0) コマンドを使ってこれを発動し、再計算を実行します。

メモ: このメッセージを使用できないプラットフォームもあります。

obj<<Bring Window To Front

選択されたウィンドウを最前面に移動する。

obj<<Close Window

指定されたオブジェクト（*obj*）を含んだウィンドウを閉じる。オブジェクトは通常、プラットフォームのウィンドウです。

obj<<Column Switcher (default column, {col 1, col 2, ...})

列スイッチャーの設定パネルをプラットフォームに追加し、変数を切り替えられるようにする。

obj<<Copy ByGroup Script

この分析を再現するスクリプト（By 変数を含む）を生成し、クリップボードにコピーする。

obj<<Copy Script

この分析を再現するスクリプトを生成し、クリップボードにコピーする。

obj<<Data Table Window

関連するデータテーブルウィンドウをアクティブにする（最前面に出す）。

obj<<Get Data Table

データテーブルへの参照を戻す。

obj<<Get Script

分析を再度実行するためのスクリプトを、式としてログに戻す。

obj<<Get Script With Data Table

分析を再現するスクリプトを、元のデータテーブルへの参照も含めて生成し、それを式としてログに戻す。

obj<<Get Timing

プラットフォームの起動にかかった時間を取得して、ログに戻す。

obj<<Get Web Support

インタラクティブ HTML で保存しようとしているディスプレイツリーのスコアを戻す。戻り値は、-1（サポートなし）、0（サポート）、または 1（サポート）のいずれかです。スコアが -1 でなければ、インタラクティブ HTML がサポートされ、<<Save Interactive HTML() が使用できます。

obj<<Get Window Position

選択されたウィンドウの位置を取得する。縦と横のサイズをリストで戻します。

obj<<Get Window Size

選択されたウィンドウのサイズをピクセルで取得する。縦と横のサイズをリストで戻します。

obj<<Ignore Platform Preferences

プラットフォームの現在の環境設定を無視する。

obj<<Journal Window

ウィンドウの内容をジャーナルに追加する。

obj<<Local Data Filter

このプラットフォームに対してのみ有効なフィルタで、データを特定のグループまたは範囲にフィルタリングする。

obj<<Maximize Window

ウィンドウを最大化する。ウィンドウの右上隅にある最大化ボタンをクリックするのと同じです。このメッセージはオプションでブール値の引数をとります。

```
// ウィンドウを最大化する
obj<<Maximize Window(1)
// ウィンドウを元に戻す
obj<<Maximize Window(0)
```

obj<<Minimize Window

ウィンドウを最小化する。ウィンドウの右上隅にある最小化ボタンをクリックするのと同じです。このメッセージはオプションでブール値の引数をとります。

```
// ウィンドウを最小化する
obj<<Minimize Window(1)
```

```
// ウィンドウを元に戻す
obj<<Minimize Window(0)
```

---

obj<<Move Window(x, y)

ウィンドウをスクリーン上の (x, y) の位置に移動する。

---

obj<<Print Window

選択されたウィンドウを印刷する。

---

obj<<Redo Analysis

同じオプションのままで、分析を再実行する。

---

obj<<Redo ByGroup Analysis

By グループを含む同じ分析をやり直す。

---

obj<<Relaunch Analysis

この分析の起動画面を表示する。

---

obj<<Relaunch ByGroup

この分析（By グループを含む）の起動画面を表示する。

---

obj<<Remove Column Switcher

プラットフォームに追加された列スイッチャーをすべて削除する。

---

obj<<Remove Local Data Filter

プラットフォームに追加されたローカルデータフィルタをすべて削除する。

---

obj<<Report

Report(obj)

プラットフォームウィンドウ内にあるレポートに対する、ディスプレイボックスへの参照を戻す。詳細については、『スクリプトガイド』の「表示ツリー」章を参照してください。

---

obj<<Report View

プラットフォームレポートの詳細を表示するかどうかを決定する。Full はすべての詳細を表示し、Summary はプラットフォームにより限定されたものだけを表示する。動作をカスタマイズする場合は、ディスプレイボックスに対して Set Summary Behavior メッセージを使用する。

obj<<Save ByGroup Script to Data Table

By 変数を含む分析を生成するスクリプトを作成し、データテーブルのテーブルプロパティとして保存する。

obj<<Save ByGroup Script to Journal

By 変数を含む分析を生成するスクリプトを作成し、スクリプトのボタンをジャーナルに追加する。

obj<<Save ByGroup Script to Script Window

By 変数を含む分析を生成するスクリプトを作成し、現在のスクリプトウィンドウに表示する。

obj<<Save Script for All Objects

オブジェクトのウィンドウ内にある分析を再度実行するためのスクリプトを、スクリプトウィンドウに保存する。

obj<<Save Script to Data Table

分析を再度実行するためのスクリプトを、関連するデータテーブルのプロパティとして保存する。

obj<<Save Script to Journal

分析を生成するスクリプトを作成し、スクリプトのボタンをジャーナルに追加する。

obj<<Save Script to Report

分析を再度実行するためのスクリプトを、レポートの最上部にあるテキストボックスに保存する。

obj<<Save Script for All Objects to Data Table

レポート中のすべてのオブジェクトのスクリプトをデータテーブルに保存する。スクリプト名を引用符で囲んで指定しない場合、プラットフォームにちなんだ名前が付けられます。

```
obj << Save Script for All Objects To Data Table("My Script")
```

obj<<Save Script to Script Window

分析を再度実行するためのスクリプトを、スクリプトウィンドウに保存する。

obj<<Scroll Window(x, y)

obj<<Scroll Window({x, y})

ウィンドウを右方向に *x* ピクセル、下方向に *y* ピクセル、スクロールする。負の座標の場合は左方向および上方向になります。座標が { } で囲まれたリストとして指定されている場合、その座標は絶対座標になります。ウィンドウは、左側から *x* ピクセル、最上部から *y* ピクセルの点にスクロールします。

obj<<SendToReport

レポートの外観をカスタマイズするため、Dispatch 関数とともに使用する。

obj<<SendToByGroup

プラットフォームを開いたり、プラットフォームオプションをオンにするメッセージを、By グループの各水準に送る。

obj<<Show Window(0|1)

1 はウィンドウを表示（最前面に移動）し、0 はウィンドウを非表示にする。ウィンドウが（Windows で）最小化されているか（Macintosh で）固定されている場合は、ウィンドウを表示すると通常の状態に戻され、最前面に表示されます。

obj<<Size Window(x, y)

ウィンドウのサイズを、幅 *x* ピクセル、高さ *y* ピクセルに変更する。

obj<<Title

プラットフォームのタイトルを戻す。

obj<<Top Report

レポート内の先頭のディスプレイボックスに対する参照を戻す。1 つのウィンドウに複数のプラットフォームのレポートが表示されている場合や、By グループごとに分析を行った場合に使うと便利です。

obj<<View Web XML

インタラクティブ HTML レポートを作成するのに使用した XML を戻す。XML コードはログに表示されます。

obj<<Zoom Window

内容がすべて表示されるようにウィンドウのサイズを拡大する。

応答のスクリーニング

obj<<Get PValues

「PValues」テーブルへの参照を戻す。

obj<<Save PValues

p 値を含むデータテーブルを新規に作成する。

obj<<Save Compare Means

出力データテーブルに平均の比較を保存する。

obj<<Save Mean

平均の比較に関するデータテーブルを新規に作成する。

obj<<Save Outlier Indicator

各あてはめに対して、外れ値を示す指示変数を保存する。

obj<<Save Std Residuals

各あてはめに対して、残差の計算式を保存する。

obj<<Select Columns

このテーブルで選択されている行に対応する列を、元のテーブル内で選択する。

表の作成

obj<<Display Column Width(<Data Column(Column Table(n),"colname_path"), Row Label(Row Table(n), "colname_path")>, <width>)

「表の作成」テーブル内の列の表示幅（ピクセル）を戻す、または設定する。

引数

data column テーブルの本体の列を定義する場合は、Column Table を列参照とともに使用する。

row label 行ラベル領域内の列を定義する場合は、Row Table を見出しとともに使用する。

colname_path オプションの Column Table または Row Table、列のパスを追跡する一連の列見出し。

メモ：参照先のテーブルが最初のテーブルの場合、列テーブルまたは行テーブルを省略できます。

width 列の幅をピクセルで指定する。

例

```
dt = Open( "$SAMPLE_DATA/Car Poll.jmp" );
obj = dt << Tabulate(
   Add Table(
      Column Table(
         Grouping Columns( :性別 , :Name(" 既婚 / 未婚 ") ),
         Analysis Columns( :年齢 ),
         Statistics( Sum, "% of Total" )
      ),
      Row Table( Grouping Columns( :タイプ ) ),
      Row Table( Grouping Columns( :生産国 , :サイズ ) )
   )
```

```
);
Wait( 3 ); // デモ用
obj << Display Column Width( Row Label( Row Table( 2 ), "生産国" ), 150 );
Wait( 3 ); // デモ用
obj << Display Column Width(
   Data Column(
      Column Table( 1 ),
      "性別",
      "女性",
      "既婚/未婚",
      "既婚",
      "年齢",
      "合計"
   ),
   150
);
```

Rインテグレーションメッセージ

R接続の機能は、JSL関数としてだけではなく、オブジェクトへのメッセージとしても用意されています。R Connect()というJSL関数によって、R接続オブジェクトへの参照を取得できます。R接続オブジェクトに対して、以下で述べるメッセージを送ることができます。

rconn<<Control(Interrupt (1))

R Submit()におけるAsyncオプションに1（真）が設定されているとき、このメッセージを送ると、サブミットされたRコードの処理がただちに中止される。

rconn<<Disconnect()

R接続を解除します。

rconn<<Is Connected()

R接続がアクティブな場合は1、そうでない場合は0を戻す。

rconn<<Send("name", named_arguments)

指定のJMP変数をRに送る。

戻り値

成功した場合は0、そうでない場合は0以外

引数

name　Rに送るJMP変数の名前を示した引用符付き文字列

データテーブルへの引数

`Selected(0|1)`（オプション）ブール値。真の場合、参照先データテーブルで選択されている行のみをRに送ります。

`Excluded(0|1)`（オプション）ブール値。真の場合、参照先データテーブルで除外されている行のみをRに送ります。

`Labeled(0|1)`（オプション）ブール値。真の場合、参照先データテーブルでラベルがついている行のみをRに送ります。

`Hidden(0|1)`（オプション）ブール値。真の場合、参照先データテーブルで非表示になっている行のみをRに送ります。

`Colored(0|1)`（オプション）ブール値。真の場合、参照先データテーブルで色がついている行のみをRに送ります。

`Markered(0|1)`（オプション）ブール値。真の場合、参照先データテーブルでマーカーがついている行のみをRに送ります。

`Row States(0|1, <named arguments>)`（オプション）ブール値の引数とオプションの名前付き引数を指定する。「RowState」という名前のデータ列を追加し、参照先データテーブルの行の属性情報をRに送ります。行の属性値は、表3.2に示す値を持った個別の設定で構成されます。

表3.2 行の属性

個別の設定をまとめて追加することによって、複数の行の属性を作成できます。	Selected = 1 Excluded = 2 Labeled = 4 Hidden = 8 Colored = 16 Markered = 32
引数	`Colors(Boolean)`（オプション）ブール値。真の場合、行の色を送り、「RowStateColor」という名前のデータ列を追加します。 `Markers(Boolean)`（オプション）ブール値。真の場合、行のマーカーを送り、「RowStateMarker」という名前のデータ列を追加します。

rconn<<Send("name")

指定のJMPデータファイルをRに送る。

rconn<<Get("name")

指定の変数をRから取得する。

戻り値

指定の変数の値

引数

name Rから取得するJMP変数の名前を示した引用符付き文字列

rconn<<Get Graphics("type")

Rのグラフウィンドウに最後に出力されたグラフを、指定の形式で取得する。グラフィックオブジェクトは指定のグラフィック形式で戻されます。

戻り値

JMPピクチャーオブジェクト

引数

type Rのグラフを取得するときの変換形式。有効な形式は、"png"、"bmp"、"jpeg"、"jpg"、"tiff"、および"tif"です。

rconn<<Submit("code", named_arguments)

Rコードをサブミットする。

戻り値

成功した場合は0、そうでない場合は0以外

引数

code サブミットするRコードを示す引用符付き文字列

Expand(0|1)（オプション）ブール値。コードをサブミットする前に、Rコードに対してEval Insert()を実行します。

Echo(0|1)（オプション）ブール値。実行したRのプログラムコードを、JMPのログに出力します。デフォルト値は1（真）。

Rconn<<Submit File("filepath")

"*filepath*"によって指定されたファイルにあるプログラムを、Rでサブミットする。

引数

filepath 実行するRコードを含むファイルのパス名を示す引用符付き文字列

rconn<<Execute({ list of inputs }, { list of outputs }, "code", named_arguments)

Rに対して、第1引数のリストに指定されたJMP変数を送り、第3引数に指定されたRコードをサブミットする。第2引数のリストに指定されたR変数が、JMPに戻されます。

戻り値

成功した場合は0、そうでない場合は0以外

引数

{ list of inputs } 入力としてRに送られるJMP変数名のリスト

{ list of outputs } 出力としてRから取得されるJMP変数名のリスト

code サブミットするRコードを示す引用符付き文字列

named_arguments rconn<<Submit("code", named_arguments)（336ページ）を参照してください。

rconn<<Control(named_arguments)

Rの実行を制御する。

戻り値
なし

引数
Echo(Boolean)（オプション）ブール値。実行したRのプログラムコードを、JMPのログに出力します。

rconn<<Get Version()

現在インストールされているRのバージョンを取得する。

戻り値
Rのバージョン番号を示す行列

rconn<<JMP Name To R Name("name")

Rの命名規則に従い、JMP変数名を、対応するR変数名に変換する。

戻り値
Rでの命名規則に従った変数名を示す文字列

引数
name Rの変数名に変換するJMP変数名を示す引用符付き文字列

SASインテグレーションメッセージ

メタデータサーバーオブジェクト

metaserver<<Disconnect()

機能
メタデータサーバーとの接続を解除する。

戻り値
なし

metaserver<<Get Display Name()

機能
メタデータサーバーの表示名を取得する。

戻り値
文字列

metaserver<<Get Host Name()

機能
メタデータサーバーのホスト（コンピュータ）名を取得する。

戻り値
文字列

metaserver<<Get Port()

機能
メタデータサーバー接続に使用されたポートを取得する。

戻り値
整数

metaserver<<Get User Identity()

機能
この接続を行っているユーザの、メタデータで定義されている ID を取得する。

戻り値
文字列

metaserver<<Get User Name()

機能
メタデータサーバー接続に使用されたユーザ名（ログイン ID）を取得する。

戻り値
文字列

SAS サーバーオブジェクト

sasconn<<Assign Libref("libref", "path", engine, engine options)

機能
この SAS サーバー接続上において、SAS ライブラリ参照を割り当てる。

戻り値
なし

引数
「JSL 関数」章の「SAS Assign Lib Refs("libref", "path", <"engine">, <"engine options">)」（208 ページ）を参照してください。

sasconn<<Cancel Submit()

機能

このサーバーに対して非同期で実行されていると考えられるSAS Submitをキャンセルする。

戻り値

実行中のサブミットが見つかりキャンセルされた場合は1、そうでない場合は0

sasconn<<Clear Log History()

機能

このサーバーのSASログ履歴を消去する。

戻り値

なし

sasconn<<Clear Output History()

機能

このサーバーのSASアウトプット履歴を消去する。

戻り値

なし

sasconn<<Connect(<named arguments>)

機能

切断されたSASサーバー接続オブジェクトの再接続を試みる。

戻り値

接続に成功したときは1、そうでなければ0

名前付き引数

名前付き引数はすべてオプションです。

UserName("name") 接続に必要なユーザ名を示す引用符付き文字列

Password("password")（オプション）接続のためのパスワードを示す引用符付き文字列

Prompt(Always|Never|IfNeeded) キーワード。Always では、接続を試みる前に常にプロンプトを表示します。Never では、接続の試みに失敗した場合でもプロンプトを表示しません（失敗しても、ただ単にログにエラーメッセージが送られるだけ）。IfNeeded（デフォルト値）では、与えられたパラメータで接続に失敗した場合（または与えられた認証情報では接続できない場合）にプロンプトを表示します。

sasconn<<Deassign Libref("libref")

機能

このSASサーバー接続上において、SASライブラリ参照の割り当てを解除する。

戻り値

なし

引数

libref ライブラリの参照名を示す引用符付き文字列

---

sasconn<<Disconnect()

機能

この SAS サーバー接続を切断する。

戻り値

なし

---

sasconn<Does Module Exist("moduleName")

機能

指定の SAS モジュールが、接続された SAS サーバーに存在するかどうか判断する。これは、特定の SAS 製品がインストールされているかどうかを調べるのに便利です。モジュールの存在を特定するのに、SAS データステップ関数 MODEXIST が使用されます。MODEXIST は SAS 9.2 から導入されたため、SAS 9.2 より前のバージョンでの SAS 接続に対して呼び出された場合は、例外をスローします。

戻り値

指定のモジュールが存在する場合は 1、そうでない場合は 0

引数

moduleName 存在を確認する SAS モジュール。引用符付きで指定します。拡張子はつけません。

---

sasconn<<Export Data(dt, "library", "dataset", <named arguments>)

機能

アクティブな SAS サーバー上の指定されたライブラリ内に、JMP データテーブルを SAS データセットとして書き出す。

戻り値

データテーブルが正常に書き出されたときは 1、そうでなければ 0

引数

「JSL 関数」章の「SAS Export Data(dt, "library", "dataset", <named_arguments>)」（210 ページ）を参照してください。

---

sasconn<<Get Data Sets("libref")

機能

この SAS サーバー接続から、SAS ライブラリ内に定義されているデータセットのリストを戻す。

戻り値

文字列のリスト

引数

libref SAS ライブラリ参照名または表示ライブラリ名を示す引用符付き文字列。引数で指定されたライブラリ内にある SAS データセットがリストで戻されます。

sasconn<<Get Error Count()

機能

前回の SAS Submit で生じたエラーの数を取得する。

戻り値

整数

sasconn<<Get File("source", "dest")

機能

この SAS サーバー接続からファイルをダウンロードする。

戻り値

なし

引数

「JSL 関数」章の「SAS Get File("source", "dest", "encoding")」（211 ページ）を参照してください。

sasconn<<Get File Names("fileref")

機能

このサーバー接続から、与えられたファイル参照（"*fileref*"）にあるファイル名のリストを取得する。

戻り値

文字列のリスト

引数

`fileref` ファイル参照名を示す引用符付き文字列。指定されたファイル参照の配下にあるファイルの名前が取得されます。

sasconn<<Get File Names In Path("path")

機能

この SAS サーバー接続から、与えられたパス（*"path"*）にあるファイル名のリストを取得する。

戻り値

文字列のリスト

引数

`path` ファイル名を取得するサーバー上のディレクトリパスを示した引用符付き文字列

sasconn<<Get File Refs()

機能

この SAS サーバー接続から、現在定義されている SAS ファイル参照のリストを取得する。

戻り値

文字列のリスト

sasconn<<Get Librefs(<named arguments>)

機能

このSASサーバー接続から、現在定義されているSASライブラリ参照のリストを取得する。

戻り値

文字列のリスト

引数

「JSL関数」章の「SAS Get Lib Refs(<named arguments>)」（212ページ）を参照してください。

sasconn<<Get Log()

機能

このサーバー接続から、前回のSAS SubmitのSASログを取得する。

戻り値

文字列

sasconn<<Get Option Name()

機能

SASに対し、SASのオプション変数の値を照会する。

戻り値

文字列

例

次のスクリプトは、指定された変数を反復処理し、変数値を出力します。

```
option_names = sasconn << Get Option Names();
For(i=1, i <= N Items(option_names), i++,
   option_value = sasconn << Get Option Value (option_names[i]);
   output = option_names[i] || "=" || char(option_value) || "/!n";
   Write(output);
);
```

sasconn<<Get Output()

機能

このSASサーバーオブジェクトへ最後に送信したSASコードの出力リストを戻す。

戻り値

文字列

sasconn<<Get Results()

機能

前回のSASサブミットの結果をスクリプト可能なオブジェクトとして取得する。これにより、結果を柔軟に処理できるようになります。

戻り値

スクリプト可能なSAS実行結果オブジェクト

sasconn<<Get Submit Status()

機能

このサーバーに対して非同期で実行されていると考えられるSAS Submitの現在の状態を取得する。

戻り値

サブミットが開始されていない場合は1、サブミットが実行中の場合は2、サブミットがキャンセルされた場合は3、サブミットが正常に完了した場合は10、サブミットにエラーが生じた場合は11

sasconn<<Get Var Info("libref", "dataset", <"password">)

機能

指定のSASデータセットの変数に関する情報を戻す。

引数

`libref` 解除するライブラリの参照名を示した引用符付き文字列

`dataset` 変数名を取得するデータセットの名前を示す引用符付き文字列

`Password("password")`（オプション）接続のためのパスワードを示す引用符付き文字列

sasconn<<Get Var Names("libref", "dataset", <named arguments>)

機能

このSASサーバー接続から、指定のデータセットに含まれている変数の変数名を取得する。

戻り値

文字列のリスト

引数

「JSL関数」章の「SAS Get Var Names(string, <"dataset">, <password("password")>)」（213ページ）を参照してください。

sasconn<<Get Version(<"long">)

機能

SASバージョンを"9.1"や"9.2"などの文字列で戻す。

戻り値

SASバージョンを示す文字列

引数

`long`（オプション）SASの長いバージョン名を戻すよう指定する引用符付きキーワード。長いバージョン名はSYSVLONG SASマクロに対応します。たとえば、"9.02.02M0P01152009"

sasconn<<Get Work Folder()

機能

このサーバーの WORK ライブラリに対応するフォルダの完全パスを戻す。

戻り値

WORK フォルダのパスを示す文字列

sasconn<<Import Data("library", "dataset", <named arguments>)

機能

この SAS サーバー接続から、JMP データテーブルに SAS データセットを読み込む。

戻り値

JMP データテーブルオブジェクト

引数

「JSL 関数」章の「SAS Import Data(string, <"dataset">, <named arguments>)」（214 ページ）を参照してください。

sasconn<<Import Query("sqlquery", <named arguments>)

機能

要求された SQL クエリをこの SAS サーバー接続で実行し、その結果を JMP データテーブルに読み込む。

戻り値

JMP データテーブルオブジェクト

引数

「JSL 関数」章の「SAS Import Query("sqlquery", <named arguments>)」（215 ページ）を参照してください。

sasconn<<Is Connected()

機能

この SAS サーバーオブジェクトが現在 SAS に接続されているかどうかを判断する。

戻り値

SAS サーバーオブジェクト（*sasconn*）が接続されていれば 1、そうでなければ 0

sasconn<<Is Product Available("productName")

機能

SAS 接続されたセッションにおいて、指定の SAS 製品がライセンス供与され、インストールされているかどうかを特定する。製品のライセンスおよびインストールの状態を特定するには、SAS データステップ関数の SYSPROD および MODEXIST が使用されます。

戻り値

指定の製品がライセンス供与されている場合は 1、されていない場合は 0、指定の製品が SAS によって認識されない場合は -1 を戻す。また、指定の製品が JMP ではインストールの状態を特定できない製品である場合は、例外を戻します。

引数

`productName` ライセンス状態を確認する SAS 製品。引用符付きで指定します。製品名には、接頭部の「SAS/」をつけてもつけなくてもかまいません。

ノート

MODEXIST 関数は SAS 9.2 での新機能です。SAS 9.1.3 の場合、この関数はライセンスのみを確認し、インストールの状態は確認しません。つまり SAS 9.1.3 では、この関数と `Is Product Licensed()` に違いがありません。

`sasconn<<Is Product Licensed("productName")`

機能

指定の SAS 製品が、SAS 接続されたセッションにおいてライセンス供与されているかどうかを特定する。製品のライセンスの状態を特定するには、SAS データステップ関数の SYSPROD が使用されます。

戻り値

指定の製品がライセンス供与されている場合は 1、されていない場合は 0、指定の製品が SAS によって認識されない場合は -1 を戻す。

引数

`productName` ライセンス状態を確認する SAS 製品。引用符付きで指定します。製品名には、接頭部の「SAS/」をつけてもつけなくてもかまいません。

`sasconn<<Kill Session(<n>)`

機能

SAS 接続を即座に終了する。

戻り値

なし

引数

`n` （オプション）数字。通常終了するために n 秒待機してから、SAS 接続を即座に終了する。

`sasconn<<Load Text File("path", <named arguments>)`

機能

アクティブな SAS サーバー接続から、パス（"*path*"）で指定されたファイルをダウンロードし、その内容を文字列で取得する。

戻り値

文字列

引数

「JSL 関数」章の「SAS Load Text File("path")」（217 ページ）を参照してください。

sasconn<<Open Log Window()

機能

このサーバーの「SAS ログ」ウィンドウを開く（または最前面に表示する）。

戻り値

なし

sasconn<<Open Output Window()

機能

このサーバーの「SAS アウトプット」ウィンドウを開く（または最前面に表示する）。

戻り値

なし

sasconn<<Open SAS Results()

機能

前回の SAS Submit の結果を開く。非同期の SAS Submit や、SAS Submit の OnSubmitComplete オプションを使用すれば、サブミットの結果を条件に応じて開くことができます。

戻り値

なし

sasconn<<Open Submit Results()

機能

前回の SAS Submit コマンドの結果をすべて開く。

戻り値

なし

sasconn<<Send File("source", "dest")

機能

この SAS サーバー接続にファイルをアップロードする。

戻り値

なし

引数

「JSL 関数」章の「SAS Send File("source", "dest", "encoding")」（217 ページ）を参照してください。

sasconn<<Submit(sasCode, <named arguments>)

機能

この SAS サーバー接続に SAS コードをサブミットする。

戻り値

なし

引数

「JSL 関数」章の「SAS Submit("sasCode", <named arguments>)」（218 ページ）を参照してください。

sasconn<<Submit File("filename", <named arguments>)

機能

この SAS サーバー接続にファイルにある SAS コードをサブミットする。

戻り値

なし

引数

「JSL 関数」章の「SAS Submit File("filename", <named arguments>)」（219 ページ）を参照してください。

ストアドプロセスオブジェクト

stp<<Begin Run(<named arguments>)

機能

このストアドプロセスのバックグラウンドでの実行を開始する。このメッセージは End Run と併用します。ストアドプロシージャの完了を待つため、End Run は Begin Run よりいくらか後の時点で呼び出します。

戻り値

-1 = 実行に失敗

1 = 開始前

2 = 実行中

3 = キャンセル済み

10 = 正常に終了

11 = 終了（エラー）

引数

Run と同じだが、AutoOpenResults と NoAlerts は使用できない。どちらも、EndRun で使用可能

AutoResume(<"filename">)（オプション）引用符付き文字列。引数が指定されていない場合、ストアドプロセスの完了時にストアドプロセスの結果が自動的に開く。ファイル名が指定されている場合、ストアドプロセスのすべての結果ではなく、指定のファイルだけが開く。

AutoResumeScript("script")（オプション）ストアドプロセスの実行後、指定されたスクリプトを評価する。スクリプトが1つ以上の引数を取る関数の場合、最初 (かつ唯一) の引数として渡されたストアドプロセスオブジェクトを用いて、その関数は評価される。AutoResume および AutoResumeScript は両方を一緒に指定することはできない。

stp<<Delete Results(<named arguments>)

機能

このストアドプロセスの実行結果をすべて削除する。

戻り値

正常に削除できたときは 1、そうでなければ 0（エラーメッセージが JMP ログに送られる）

引数

NoAlerts(0|1)（オプション）ブール値。1（真）の場合、ユーザに対して確認のプロンプトが表示されないまま結果が削除される

DeleteDirectory(0|1)（オプション）ブール値。1（真）の場合は、ストアドプロセスの結果を含んだディレクトリを、結果ファイル自体と共に削除します。デフォルト値は 1（真）。

stp<<Edit Param Values()

機能

パラメータ値をインタラクティブに設定できるよう、ストアドプロセスウィンドウを起動する。

戻り値

ユーザが［**OK**］をクリックした場合は 1、［**キャンセル**］をクリックした場合は 0 を戻す。

stp<<End Run(<named arguments>)

機能

Begin Run で開始されたストアドプロセスが終了するまで、指定の時間だけ（または永久に）待機する。その間にストアドプロセスが終了すれば、結果を取得し、開く。

戻り値

-1 = 実行に失敗

1 = 開始前

2 = 実行中

3 = キャンセル済み

10 = 正常に終了

11 = 終了（エラー）

引数

AutoOpenResults(0|1)（オプション）ブール値。1（真）の場合、ストアドプロセスが *MaxWait* で指定された時間内に終了したら、結果を自動的に開く。0（偽）の場合、結果を自動的には開かない（その場合でも、Get Results メッセージによって、戻されたオブジェクトを使って手動で開くことはできる）。デフォルト値は、1（真）。

MaxWait(milliseconds)（オプション）整数。ストアドプロセスの終了を待機する最大時間をミリ秒で指定する。MaxWaitが指定されていない場合、End Runはストアドプロセスが終了するまで永久に待機する。

NoAlerts(0|1)（オプション）ブール値。1（真）の場合、メッセージボックスではなくJMPログにエラーメッセージが送信される。デフォルト値は0（偽）。

stp<<Get Metadata Id()

機能

ストアドプロセスのメタデータIDを戻す。

戻り値

文字列

stp<<Get Metadata Path()

機能

ストアドプロセスの完全メタデータパスを戻す。

戻り値

文字列

Stp<<Get Name()

機能

ストアドプロセスの名前を戻す。

戻り値

文字列

stp<<Get Param Enum Labels("name")

機能

パラメータの列挙ラベルを取得する。

戻り値

文字列のリスト

引数

name 列挙ラベルを取得するパラメータの名前を示した引用符付き文字列

stp<<Get Param Enum Values("name")

機能

パラメータに設定可能な列挙値を取得する。

戻り値

文字列のリスト

引数

name 設定可能な列挙値を取得するパラメータの名前を示した引用符付き文字列

---

stp<<Get Param Names(<named arguments>)

機能

このストアドプロセスのパラメータ名のリストを戻す。オプションで種類が指定されている場合は、その種類のパラメータ名のリストを戻します。

戻り値

文字列のリスト

引数

Visible(0|1)（オプション）ブール値。1（真）の場合、表示のパラメータのみを取得する。0（偽）の場合、非表示のパラメータのみを取得する。指定されていない場合、表示と非表示の両方のパラメータを取得する。

Modifiable(0|1)（オプション）ブール値。1（真）の場合、変更可能なパラメータのみを取得する。0（偽）の場合、変更不可なパラメータのみを取得する。指定されていない場合、変更可能と変更不可の両方のパラメータを取得する。

Required(0|1)（オプション）ブール値。1（真）の場合、エキスパートのパラメータのみを取得する。0（偽）の場合、エキスパートでないパラメータのみを取得する。指定されていない場合、エキスパートとエキスパートでないパラメータの両方を取得する。

Expert(0|1)（オプション）ブール値。1（真）の場合、エキスパートのパラメータのみを取得する。0（偽）の場合、エキスパートでないパラメータのみを取得する。指定されていない場合、エキスパートとエキスパートでないパラメータの両方を取得する。

---

stp<<Get Param Value("name")

機能

指定したパラメータの現在の値を取得する。

戻り値

文字列

引数

name 値を取得するパラメータの名前を示した引用符付き文字列

---

stp<<Get Results()

機能

このストアドプロセスの実行結果をスクリプト可能なオブジェクトとして取得する。

戻り値

スクリプト可能なSAS実行結果オブジェクト

---

stp<<Get Status()

機能

ストアドプロセスの実行ステータスを取得する。

戻り値

-1 = 実行に失敗

1 = 開始前

2 = 実行中

3 = キャンセル済み

10 = 正常に終了

11 = 終了（エラー）

stp<<Get Status Message()

機能

ストアドプロセスの失敗に関連するメッセージがあれば、それらを取得する。

戻り値

文字列

stp<<Reset Param Values()

機能

すべてのパラメータ値をメタデータ定義のデフォルト値にリセットする。

戻り値

なし

stp<<Run(<named arguments>)

機能

このストアドプロセスオブジェクトを前面で実行する。

戻り値

-1 = 実行に失敗

1 = 開始前

2 = 実行中

3 = キャンセル済み

10 = 正常に終了

11 = 終了（エラー）

引数

AutoOpenResults(0|1)（オプション）ブール値。1（真）の場合、ストアドプロセスの終了後、結果を自動的に開く。0（偽）の場合、結果は自動的には開かないが、GetResults メッセージによって戻されたオブジェクトを使って手動で開くことができる。デフォルト値は 1（真）。

UserName("username")（オプション）ストアドプロセスを実行するためのユーザ名を示す引用符付き文字列。

Password("password")（オプション）ユーザ名 *Username* のためのパスワードを示す引用符付き文字列

AuthDomain("authDomain")（オプション）与えられたログイン情報（*username*、*password*）の認証ドメインを示す引用符付き文字列。

ODSDest("dest")（オプション）ODSの出力先（HTML、PDF、Tagsets.SASReport12など）を示す引用符付き文字列。ストアドプロセスが生成するODSの出力先を決めるものです。これには、%STPBEGINを呼び出すストアドプロセスSASコードが必要。デフォルト値はHTML。

GraphicsDevice("device")（オプション）ODS結果内のグラフィックを生成する時に使用するSASグラフィックデバイスを示した引用符付き文字列。これには、%STPBEGINを呼び出すストアドプロセスSASコードが必要。デフォルト値はGIF。

ODSStyle("styleName")（オプション）結果に適用するODSスタイルを示した引用符付き文字列。これには、%STPBEGINを呼び出すストアドプロセスSASコードが必要。デフォルト値はなし。

ODSStyleSheet("cssFile")（オプション）生成されたODS結果に適用されるCSSファイルのクライアントコンピュータ上の完全パスを示した引用符付き文字列。これには、%STPBEGINを呼び出すストアドプロセスSASコードが必要。デフォルト値はなし。

NoAlerts(0|1)（オプション）ブール値。1（真）の場合、メッセージボックスではなくJMPログにエラーメッセージが送信される。デフォルト値は0（偽）。

stp<<Set Param Value("name", "value")

機能

指定のストアドプロセスパラメータの値を指定の値に設定する。

戻り値

成功したときは1、そうでなければ0（値がパラメータの制約に違反した場合など）

引数

name 値を設定するパラメータの名前を示した引用符付き文字列

value パラメータに設定する値を示した引用符付き文字列

stp<<Set Results Directory("directory")

機能

ストアドプロセスの結果の格納先とするクライアントコンピュータ上のディレクトリを設定する。

戻り値

文字列

引数

directory ストアドプロセスの実行結果を格納するディレクトリの完全パスを示した引用符付き文字列。ディレクトリはすでに存在しているか、または作成可能なものでなければならない。結果のディレクトリが指定されない場合、オペレーティングシステムに適した一時的な場所に格納される。ストアドプロセスの実行後、スクリプト可能なSASの結果オブジェクトから、このディレクトリを取得できる。

SASの結果オブジェクト

results<<Delete All Result Files()

機能

SASサブミットまたはストアドプロセスの実行によって作成されたファイルをすべて消去する。使用中の結果ファイルは削除されません。

戻り値

削除が正常に完了した場合は1、削除できなかったファイルがある場合は0

results<<Get Directory()

機能

ストアドプロセスまたはSASサブミットによって生成された結果ファイルがあるディレクトリを取得する。

戻り値

文字列

results<<Get Log()

機能

ストアドプロセスまたはSASサブミットの実行によって生成されたSASログを取得する。

戻り値

文字列

results<<Get Main Result File Name(<named arguments>)

機能

ストアドプロセスまたはSASサブミットによって生成された主要な結果ファイルの完全パスを取得する。

戻り値

文字列

引数

FullPath(0|1)（オプション）ブール値。1（真）の場合、主要な結果ファイル名は完全パスとして戻される。デフォルト値は0（偽）。

results<<Get Output()

機能

ストアドプロセスまたはSASサブミットの実行によって生成されたSASアプトプットを取得する。

戻り値

文字列

results<<Get Output Datasets()

機能

このSASの結果オブジェクトを作成したSASサブミットによって生成されたSASデータセットのリストを取得する。

戻り値

"ライブラリ名.メンバー名"という形のデータセット名のリスト

results<<Get Result File Info(<MIMEType("mime-type")>, <FullPath>)

機能

ストアドプロセスまたはSASサブミットの実行によって生成された結果ファイルに関する情報を取得する。

戻り値

2つの文字列リストのリスト。最初のリストはファイル名で、2番目のリストは最初のリストのファイルに対応するMIMEタイプのリスト。

引数

MIMEType("mime_type")（オプション）情報を取得するファイルのセットを、指定のMIMEタイプだけに限定するための引用符付き文字列。指定されない場合、生成されたすべてのファイルに関する情報が戻される。

FullPath（オプション）ブール値。1（真）の場合、各結果ファイルのファイル名は完全パスで戻される。0（偽）の場合、ファイル名のみが戻される。デフォルト値は0（偽）。

results<<Make JMP Report()

機能

ODS XMLの結果を解析し、JMPレポートを作成する。

戻り値

レポートのディスプレイボックス

results<<Open All Results()

機能

ストアドプロセスまたはSASサブミットの実行によって生成された結果をすべて開く。

戻り値

なし

results<<Open Result File("filename", <named arguments>)

機能

指定された名前の結果ファイルを開く。

戻り値

JMPデータテーブルが開いた場合はJMPデータテーブル

引数

`filename` 生成された結果のうち、開きたいファイルの名前を示した引用符付き文字列。ファイル名は、完全パスではなくファイルの名前で指定する。ファイル名（*`filename`*）に拡張子がない場合、JMP データテーブルと JSL スクリプトの両方から一致するものが検索される。両方に一致するものが見つかった場合、両方を開く。

`RunScript(0|1)`（オプション）ブール値。1（真）の場合、ファイル（*`filename`*）が JSL スクリプトなら、そのスクリプトが実行される。0（偽）の場合、ファイル（*`filename`*）が JSL であっても開くだけで実行されない

```
results<<Run Script("filename")
```

機能

結果の中から指定のファイル名の JSL ファイルを探し、見つかればそれを実行する。

戻り値

なし

引数

`filename` 生成された結果のうち、開きたい JSL ファイルの名前を示した引用符付き文字列。ファイル名（*`filename`*）は、完全パスではなくファイルの名前で指定する。拡張子 .jsl を含める必要はない。

スケジュール

「JSL 関数」章の「Schedule(n, script)」（257ページ）も参照。

スケジュール関連のアクションの詳細については、『スクリプトガイド』の「プログラミング手法」章を参照してください。

```
sch<<Clear Schedule()
```

スケジュールが設定されているイベントをすべてキャンセルする。

```
sch<<Close()
```

スケジューラを閉じる。

```
sch<<Restart()
```

スケジュール設定されたすべてのイベントの実行を停止した後で、スケジューラを再起動する。

```
sch<<Show Schedule()
```

スケジュールが設定されたすべてのイベントのリストを表示する。

sch<<Stop()

スケジュール設定されたすべてのイベントのスケジューラによる実行を停止する。

ソケット

skt<<Accept(<callback, timeout>)

機能

サーバーソケットに、接続を受け入れて、新しく接続されたソケットを戻すよう伝える。

戻り値

最大 4 項目のリスト。最初の文字列はコマンド（"accept"）。2 番目の文字列は "ok" またはエラー。3 番目の文字列は接続したコンピュータの名前を指定するもの。4 番目の文字列は、さらにメッセージを送信するためのソケットへの参照

引数

callback（オプション）データを受信する関数の名前を指定

timeout *callback* を使用する場合は、*timeout* で関数が応答するまでの待機時間を指定。サーバーソケットに対しては、しばらくの間接続が行われていなくてもサーバーをシャットダウンするべきではないので、0 を設定することもあるでしょう。

skt<<bind("localhost", port)

機能

ローカルコンピュータのポートをソケットに関連付ける。

戻り値

文字列 2 つのリスト。最初の文字列はコマンド名（"bind"）、2 番目の文字列は、成功した場合は "ok"、そうでなければエラー

引数

localhost ローカルコンピュータを指定。別のコンピュータにバインドすることはできない

port 使用するポート

skt<<Close()

機能

ソケットを閉じる。

戻り値

文字列 2 つのリスト。最初の文字列はコマンド名（"close"）、2 番目の文字列は、成功した場合は "ok"

skt<<Connect(socketname, port)

機能

リスニングソケットに接続する。

戻り値

文字列2つのリスト。最初の文字列はコマンド名（"connect"）で、2番目の文字列は、成功した場合は"ok"、そうでない場合は相手側のソケットから返されるエラー

引数

`socketname` 相手側のソケットの名前。Webサーバーに接続する場合は、（IPアドレスではなく）Webアドレスを指定する

`port` 相手側のソケットで接続に使用されるポート

skt<<GetPeerName()

機能

接続先のソケットのアドレスとポートを取得する。

戻り値

文字列4つのリスト。最初の文字列はコマンド（"getpeername"）。2番目の文字列は"ok"またはエラー。3番目がアドレスで、4番目がポート

skt<<GetSockName()

機能

接続元（こちら側）のソケットのアドレスとポートを取得する。

戻り値

文字列4つのリスト。最初の文字列はコマンド（"getsockname"）。2番目の文字列は"ok"またはエラー。3番目がアドレスで、4番目がポート

skt<<ioctl(FIONBIO, 1)

機能

ソケットのブロック動作をコントロールする。

戻り値

文字列2つのリスト。最初の文字列はコマンド名（"ioctl"）、2番目の文字列は、成功した場合は"ok"、そうでなければエラー

引数

`FIONBIO, 1` `FIONBIO`はNon-Blocking I/O(非ブロッキング入出力)を意味する。1はこの機能および引数をオンにする

skt<<Listen()

機能

接続をlisten状態にするようサーバーに伝える。

戻り値

文字列2つのリスト。最初の文字列はコマンド名（"listen"）で、2番目の文字列は、成功した場合は"ok"、そうでなければエラー

skt<<recv(n, <callback, timeout>)

skt<<recvfrom(n, <callback, timeout>)

機能

相手側のソケットからストリームメッセージ（recv）またはデータグラムメッセージ（recvfrom）のいずれかを受信する。2 つのオプションの引数が使用された場合、データは即座に受信されず、*callback* 関数が呼ばれたときに受信されます。

戻り値

文字列 3 つのリスト。最初の文字列はコマンド名（"recv" または "recvto"）。2 番目は、成功した場合は "ok"、そうでなければエラー。3 番目の文字列は受信したデータ。callback 関数が使用された場合は、4 番目の要素として元の recv または recvfrom メッセージで使用されたソケットを戻す

引数

n 相手側のソケットから受信するバイト数を指定
callback（オプション）データを受信する関数の名前を指定
timeout *callback* を使用する場合は、*timeout* で関数が応答するまでの待機時間を指定。

skt<<Send(stream)

skt<<SendTo(dgram)

機能

引数内のデータを、相手側のソケットに送る。Send はストリームを送り、Sendto はデータグラムを送ります。

戻り値

文字列 3 つのリスト。最初の文字列はコマンド名（"send" または "sendto"）。2 番目は、成功した場合は "ok"、そうでなければエラー。3 番目の文字列は送信できなかったストリームの一部、または、すべてのデータが正常に送信された場合は空白

引数

stream 相手側のソケットに送信するコマンド
dgram 相手側のソケットに送信するコマンド

ノート

引数にバイナリデータが必要な場合があります。JMP では、印刷できない ASCII 文字を波形符号（~）と 16 進数で示します。たとえば、次のようになります。

```
skt<<send(("GET / HTTP/1.0~0d~0a~0d~0a");
```

これは、HTTP サーバーに GET 要求を送信します。

SQL

obj<<Custom SQL(sql)

クエリーをカスタム SQL クエリーに変更し、SQL を設定する。

obj<<Generate SQL

クエリーを実行するときに生成される SQL を戻す。

obj<<Modify

クエリーをクエリービルダで開く。

obj<<PostQueryScript(script_as_text)

クエリーが実行された後に実行される JSL スクリプトを設定する。

obj<<Query Name(<newName>)

クエリーの名前を取得（引数 newName を省略）または設定（引数 newName を指定）する。クエリーの名前は、そのクエリーを実行した結果のデータテーブル名に使用される。

obj<<Run(<private|invisible>, <UpdateTable(table)>, <OnRunComplete(script)>, <OnRunCanceled(script)>, <OnError(script)>

説明

［クエリービルダー］環境設定の［可能な限りクエリーをバックグラウンドで実行］の選択にしたがって、SQL クエリーをバックグラウンドまたはフォアグラウンドで実行する。

戻り値

クエリーがバックグラウンドで実行された場合はなし。クエリーがフォアグラウンドで実行された場合はデータテーブル。

引数

private（オプション）クエリーにより生成されたデータテーブルをデータテーブルウィンドウに表示せずに開く。

invisible（オプション）クエリーが生成したデータテーブルを非表示にする。クエリーの結果を非表示にして、後に続くクエリーで使用する場合、この引数を使用してください。このデータテーブルは、ホームウィンドウの「ウィンドウリスト」または［ウィンドウ］>［再表示］のリストに表示されます。

UpdateTable（オプション）指定されたデータテーブルを更新する。クエリーはフォアグラウンドで実行される。

OnRunComplete（オプション）クエリーが完了した後に実行するスクリプトを指定する。

OnRunCanceled（オプション）ユーザがクエリーをキャンセルした後に実行するスクリプトを指定する。

OnError（オプション）エラーが起こったときに実行するスクリプトを指定する。

ノート

バックグラウンドクエリーの結果のデータテーブルを取得するには、オプションの OnRunComplete 引数を使用します。クエリーが完了したときに実行するスクリプトを記述し、さらに結果のデータテーブルへのデータテーブル参照を割り当てます。または、データテーブルを最初の引数として受け入れる関数に引数として渡すこともできます。クエリーが完了した後、その関数が呼び出されます。

例

次の例は、クエリービルダーで保存したクエリーを開きます。クエリーはプライベートに（クエリービルダーを開かずに）開きます。クエリーが実行され、結果のデータテーブルが開きます。

```
query = Open( "c:¥My Data¥Movies.jmpquery", private);
dt = query << Run();
```

スクリプトに .jmpquery ファイルを指定して、このクエリーをバックグラウンドで実行するには、<<Run Background メッセージを使用します。

```
query = Include( "C:/Queries/movies.jmpquery");
query <<Run Background();
```

次の例は、データベースにクエリーを送り、結果のデータテーブルを開き、データテーブルの行数をログに出力します。

```
confirmation = Function( {dtResult},
   Write( "\!N クエリーの結果の行数： ", N Rows( dtResult ) )
);
query = New SQL Query(
   Connection(
      "ODBC:DSN=SQL
   Databases;APP=MYAPP;TrustedConnection=yes;WSID=D79255;DATABASE=SQB;"
   ),
   QueryName( "movies_to_update" ),
   Select( Column( "YearMade", "t1" ), Column( "Rating", "t1" ) ),
   From( Table( "g6_Movies", Schema( "SQB" ), Alias( "t1" ) ) ),

);
query << Run( OnRunComplete( confirmation ) );
```

Run Background(<OnRunComplete(script), <private|invisible>>, <OnRunCanceled(script)>, <OnError(script)>

説明

SQL クエリーをバックグラウンドで実行する。実行中のクエリーは表示されません。

戻り値

なし（または、OnRunComplete が含まれている場合はデータテーブルオブジェクト）。

引数

OnRunComplete（オプション）クエリーが完了した後に実行するスクリプトを指定する。結果のデータテーブルを取得するには、OnRunComplete を含めます。

private（オプション）結果のデータテーブルを開かない。必ず OnRunComplete とともに使用します。バックグラウンドで実行するクエリーに private を指定すると、データテーブルは開かれたうえで非表示（invisible）になります。

invisible（オプション）データテーブルを非表示にする。クエリーの結果を非表示にして、後に続くクエリーで使用する場合、この引数を使用してください。このデータテーブルは、ホームウィンドウの「ウィンドウリスト」または［ウィンドウ］>［再表示］のリストに表示されます。

OnRunCanceled（オプション）ユーザがクエリーをキャンセルした後に実行するスクリプトを指定する。

OnError（オプション）エラーが起こったときに実行するスクリプトを指定する。

ノート

SAS クエリー以外のクエリーはすべてバックグラウンドで実行されます。これは、クエリービルダー環境設定の［可能な限りクエリーをバックグラウンドで実行］（デフォルトで選択されている）に基づいています。SAS クエリーの場合、Run Background() は無視されます。

スクリプトに .jmpquery ファイルを指定して、このクエリーをバックグラウンドで実行するには、<<Run Background メッセージを使用します。

```
query = Include( "C:/Queries/movies.jmpquery");
query <<Run Background();
```

Run Foreground(<OnRunComplete(script), <private|invisible>>, <OnRunCanceled(script)>, <OnError(script)>

説明

SQL クエリーをフォアグラウンドで実行する。

戻り値

クエリーが完了したときに開くデータテーブル。

引数の詳細については、「Run Background(<OnRunComplete(script), <private|invisible>>, <OnRunCanceled(script)>, <OnError(script)>」（360ページ）を参照してください。

---

obj<<Save

クエリーを、それが関連付けられているファイルに保存する。関連付けられているファイルがまだない場合、保存に失敗します。

---

obj<<Save As(path, <ReplaceExisting(Boolean))

クエリーを、指定されたファイルに保存する。ファイルがすでに存在している場合、Replace Existing が真でなければ保存に失敗します。

その他のオブジェクト

Zip アーカイブ

---

list = za<<dir

メンバー名のリストを戻す。

---

data = za<<read(membername, <format(blob)>)

メンバーデータ全体を含む文字列を戻す。オプションの引数が指定された場合は BLOB を戻します。

ノート

OPEN 関数で URL を指定して、zip データを開いた場合、JMP は、そのリモートファイルをローカルディスクにコピーします。zip データが使用されなくなると、ローカルのファイルは削除されます。

actualname = za<<write(membername, textstring|blobdata)

zip アーカイブのメンバファイルにテキストまたは BLOB を書き込む。membername に指定したファイルが現在の zip ファイル内にない場合、membername と同じ名前の actualname が戻されます。既存のメンバーファイルが上書きされないように、メンバーファイル名は変更され、実際に使用される名前が戻されます。たとえば、次のようになります。

```
name1 = za<<write(name, blobdata)
```

ジャーナル

jnl<<Save HTML(<"path">, <"format">)

ジャーナルを HTML として保存する。

path 保存する HTML ファイルのパスを引用符で囲んだ文字列（例 : "c:¥myFile.html"）。

format グラフィックファイルの形式。引用符付きで指定します。JPG、PNG、TIFF の各形式がサポートされています。グラフィックは gfx という名前のサブディレクトリに保存されます。

jnl<<Save RTF(<"path">, <"format">)

ジャーナルを RTF ファイルとして保存する。

引数

path 保存する RTF ファイルのパスを引用符で囲んだ文字列（例 : "c:¥myFile.rtf"）。

format 埋め込みグラフィックのファイル形式。引用符付きで指定します。JPG、PNG、TIF の各形式がサポートされています。

jnl<<Save PDF(<"path">, <Show Page Setup(0|1)>, <Portrait(0|1)>)

ジャーナルを PDF ファイルとして保存する。

引数

path 保存する PDF ファイルのパスを引用符で囲んだ文字列（例 : "c:¥myFile.pdf"）。

Show Page Setup 1（真）に設定すると、「ページ設定」ウィンドウが開く。このウィンドウで、余白、倍率、その他のページレイアウトオプションを変更できます。

Portrait 用紙の向きを縦または横のいずれかに設定する。ここでの設定内容は、Show Page Setup で表示されるウィンドウでユーザが選択した内容よりも優先されます。

付録 A

JMP クエリーで使用可能な SQL 関数

JSL 関数の Query() は、選択したテーブルに対して SQL クエリーを実行し、データをデータテーブルに書き出します。次の例では、Big Class.jmp に t1 という別名を割り当て、t1 テーブルから **name**、**age**（14 歳以上）、**height** を選択します。

```
dt = Open( "$SAMPLE_DATA/Big Class.jmp" );
Query( Table( dt, "t1" ),
       "SELECT t1.名前 , t1.年齢 , t1.\!"身長 ( インチ )\!"  FROM t1
       WHERE t1.年齢 > 13" );
```

クエリーでは、SQL の関数を使用できます。たとえば、SELECT CURRENT_TIMESTAMP は、現在の日時（UTC/GMT）を SQLite の日付時間文字列として戻します。

```
Query( Scalar, "SELECT CURRENT_TIMESTAMP;" );
```

この付録には、SQL クエリーで使用できる数値関数、日付時間関数、文字列関数、SQL システム関数、集計関数の一覧を示します。ネイティブの SQLite 関数には、該当欄に「○」と記します。詳細については、SQLite のオンラインマニュアル https://www.sqlite.org/lang.html を参照してください。

SQL の数値関数

数値関数	SQLite ネイティブ	説明
ABS(*number*)		指定された数値（number）の絶対値を戻す。
ACOS(*cosine*)		指定された余弦（cosine）の角度をラジアンで戻す。
ASIN(*sin*)		指定された正弦（sin）の角度をラジアンで戻す。
ATAN(*tangent*)		指定された正接（tangent）の角度をラジアンで戻す。
ATAN2(*x*, *y*)		逆正接関数（引数を 2 つとる）。
CEILING(*number*) CEIL(*number*)		指定された数値（number）より大きい最小の整数を戻す。
COS(*radians*)		指定された角度（radians）の余弦を戻す。
COT(*radians*)		指定された角度（radians）の余接を戻す。
DEGREES(*radians*)		角度（radians）をラジアンから度に変換する。
EXP(*number*)		定数 **e** を、指定された値（number）でべき乗した結果を戻す。
FLOOR(*number*)		指定された数値（number）より小さい最大の整数を戻す。
LN(*number*) LOG(*number*)		指定された数値（number）の自然対数を戻す。
LOG10(*number*)		指定された数値（number）の常用対数を戻す。
MAX(*n1*, *n2*, ...)	○	指定された数値のうち、最大値を戻す。2 つ以上の数値を指定する必要があります。
MIN(*n1*, *n2*, ...)	○	指定された数値のうち、最小値を戻す。2 つ以上の数値を指定する必要があります。
MOD(*dividend*, *divisor*)		被除数（*dividend*）を除数（*divisor*）で割った余りを戻す。小数部のある数を指定した場合は、小数点以下を切り捨てて整数にしたうえで、計算を実行します。
PI()		定数 pi（π）の値を戻す。
POWER(*number*, *power*) POW(*number*, *power*)		数値（*number*）を、指定された値（*power*）でべき乗した結果を戻す。
RADIANS(*degrees*)		角度（degrees）をラジアンに変換する。

数値関数	SQLite ネイティブ	説明
RANDOM() RANDOM(*max*) RANDOM(*min*, *max*)		乱数を戻す。RANDOM()は、0～1の値を戻します。RANDOM(max)は、0～*max*の値を戻します。RANDOM(min, max)は、*min*～*max*の値を戻します。この関数は、JSL関数のRandom Uniform()と同じです。シード値は、JSL関数のRandom Reset()を使って制御することができます。RANDOMは、RANDと略記することもできます。
RANDOMBLOB(*length*)	○	NバイトのBLOBの乱数を戻す。詳細については、SQLiteのオンラインマニュアルhttps://www.sqlite.org/lang.htmlを参照してください。
ROUND(*number*, *<precision>*)		数値（*number*）を、指定された小数点以下の桁数（*precision*）になるように四捨五入する。*precision*のデフォルト値は0です。また、*precision*には負の値を指定することもできます。
SIGN(*number*)		*number*が正の数である場合は1を、*number*が負の数である場合は-1を、*number*が0である場合は0を戻す。
SIN(*radians*)		指定された角度（radians）の正弦を戻す。
SQRT(*number*)		*number*の平方根を戻す。
TAN(*radians*)		指定された角度（radians）の正接を戻す。
TRUNCATE(*number*, *<precision>*)		数値（*number*）を、指定された小数点以下の桁数（*precision*）になるように切り捨てる。*precision*のデフォルト値は0です。また、*precision*には負の値を指定することもできます。TRUNCATE()は、TRUNC()と略記することもできます。

SQLの日付時間関数

JMPのクエリーで日付時間関数を使用する場合は、SQLエンジン（SQLite）とJMPの日付格納形式が異なるため、注意が必要です。SQLiteは、日付時間値を文字列として格納しますが、JMPは、1904年1月1日からの秒数で格納します。テーブルに日付時間値の列がある場合は、この変換は自動的に行われますが、日付時間値を戻す関数を使用した場合は、必要に応じてJMPで変換する必要があります。

CURRENT_TIMESTAMP関数を例にとって説明します。CURRENT_TIMESTAMPはビルトインのSQLite関数で、現在の日時（UTC/GMT）をSQLiteの日付時間文字列として戻します。

```
Query( Scalar, "SELECT CURRENT_TIMESTAMP;" );
```

この式は、次の値を戻します。

```
"2016-02-16 15:44:42"
```

この文字列を解析し、JMP の日付時間値として戻すこともできますが、その手間を省くためには、以下のように CURRENT_TIMESTAMP 関数を JMPDATE() 関数に挿入します。

```
Query( Scalar, "SELECT JMPDATE( CURRENT_TIMESTAMP );" );
```

この式は、次の値を戻します。

3538482531

これは、JMP の日付時間値（表示形式を設定していない状態）です。SQLite の日付時間文字列を、別の SQL 日付時間関数に渡す場合は、自動的に JMP の日付時間値に変換されるので、JMPDate() を使用する必要はありません。以下はその例です。

```
Query( Scalar, "SELECT EXTRACT('YEAR', CURRENT_TIMESTAMP);" );
```

SQLite ネイティブの日付時間関数（date()、time()、datetime()、julianday()、strftime()）は、JMP の日付時間値と互換でないため、JMP クエリーではこれらの関数を使用しないでください。

日付時間関数	SQLite ネイティブ	説明
CURRENT_DATE	○	現在の日付（UTC/GMT）を、SQLite の日付時間文字列として戻す。
CURRENT_TIME	○	現在の時間（UTC/GMT）を、SQLite の日付時間文字列として戻す。
CURRENT_TIMESTAMP	○	現在の日時（UTC/GMT）を、SQLite の日付時間文字列として戻す。
DATEDIFF(*date1*, *date2*, *interval*, *<alignment = "Start">*)		2 つの日付の差を計算する（単位を *interval*、基準を *alignment* で指定する）。この関数は、JSL 関数の Date Difference() と同じように動作します。*interval* には、"Year"、"Quarter"、"Month"、"Week"、"Day"、"Hour"、"Minute"、"Second" のいずれかを指定できます。*alignment* には、"Start"、"Actual"、または "Fractional" を指定できます。*alignment* が指定されていない場合、"Start" が使用されます。

日付時間関数	SQLite ネイティブ	説明	
EXTRACT(*datepart*, *datetime*, *<use_locale = 1>)*		日付値または日付時間値の一部を抽出する。*datetime*には、JMPの日付時間値か、SQLiteの日付時間文字列を指定できます。*use_locale*（オプション）は、日付時間値の一部を名前で戻す関数において（月名（"MonthName"）や曜日（"DayName"）など）、現在の言語と英語のどちらを使用するかを指定します。*datepart*には、以下の値を指定できます。	
		"Year"	年を数字で戻す。
		"Month"	月を数字で戻す（1-12）。
		"MonthName"	月名を、現在の言語（*use_locale = 1*）または英語（*use_locale = 0*）で戻す。
		"Mon", "MMM"	月名を略称で戻す。
		"Day"	日にち（月単位）を戻す（1-31）。
		"DayName"	曜日を戻す。
		"DayOfWeek"	曜日を数字で戻す（1-7）。
		"DayOfYear"	日にち（年単位）を戻す（1-366）。
		"Quarter"	四半期の値を戻す（1-4）。
		"Hour"	時間を戻す（0-23）。
		"Minute"	分を戻す（0-59）。
		"Second"	秒を戻す（小数点を含む）。
		"Date"	日付の部分だけ抽出して、JMPの日付時間値として戻す。
		"Time"	時間の部分だけ抽出して、JMPの日付時間値として戻す。
JMPDATE(*SQLite time string*)		SQLiteの日付時間文字列を、JMPの日付時間値に変換する。	
NOW()		TODAY() と同じ。	
TODAY()		現在のJMP日付時間値を**ローカルの**時間で戻す（JMPのToday()関数と同じ）。	

SQL の文字列関数

関数	SQLite ネイティブ	説明
HEX(*binary*)	○	BLOB を 16 進数文字列に変換するビルトインの SQLite 関数。RANDOMBLOB() 関数と組み合わせて使用すると便利です。
JLEFT(*string*, *len*, *<pad>*)		JSL の Left() 関数と同じ。文字列（*string*）の先頭から *len* 文字を抽出して戻す。*pad* が指定され、文字列（*string*）の長さが *len* より短い場合は、*pad* を末尾に追加して、長さを *len* にして戻します）。
JRIGHT(*string*, *len*, *<pad>*)		JSL の Right() 関数と同じ。文字列（*string*）の末尾から、*len* 文字を抽出して戻す。*pad* が指定され、文字列（*string*）の長さが *len* より短い場合は、*pad* を先頭に追加して、長さを *len* にして戻します）。
LENGTH(*string*)	○	ANSI 標準の CHAR_LENGTH() に対応する SQLite 関数。文字列（string）の文字数を戻します。
LOCATE(*string1*, *string2*) POSITION(*string1*, *string2*)		文字列 *string2* の中から文字列 *string1* を探し、その開始位置を戻す（先頭位置を 1 とする）。見つからなかった場合は、0 を戻す。
LOWER(*string*)		文字列（*string*）に含まれる大文字をすべて小文字に変換して戻す。
LTRIM(*string*, *<trimchars>*)	○	文字列（*string*）の先頭が、*trimchars* に指定された文字のいずれかと一致する場合、これを削除した結果を戻す。*trimchars* を省略した場合は、スペースが削除される。
PRINTF(*format*, *<arg1, ..., argN>*)	○	表記と引数を指定して、文字列を生成する。詳細については、SQLite のオンラインマニュアル https://www.sqlite.org/lang.html を参照してください。
REPLACE(*string*, *find*, *replace*)	○	文字列（*string*）の中から文字列（*find*）をすべて検索し、文字列（*replace*）で置き換える。*replace* に数値を指定した場合は、文字列に変換されます。
REVERSE(*string*)		文字列（*string*）を反転して戻す。

関数	SQLite ネイティブ	説明
RTRIM(*string*, *<trimchars>*)	○	文字列（*string*）の末尾が、*trimchars*に指定された文字のいずれかと一致する場合、これを削除した結果を戻す。*trimchars*を省略した場合は、スペースが削除される。
SPACE(*length*)		*length*個のスペースからなる文字列を戻す。
SUBSTR(*string*, *start*, *<length>*)	○	文字列（*string*）の指定された位置（*start*）から*length*で指定した数の文字を戻す（先頭位置を1とする）。*length*を省略した場合は、指定された点（*start*）から末尾までが戻されます。
TRIM(*string*, *<trimchars>*)		文字列（*string*）の末尾が、*trimchars*に指定された文字のいずれかと一致する場合、これを削除した結果を戻す。*trimchars*を省略した場合は、スペースが削除される。
UPPER(*string*)		文字列（*string*）に含まれる小文字をすべて大文字に変換して戻す。

SQLのシステム関数

関数	SQLite	説明
COALESCE(*arg1*,..., *argN*)	○	渡された引数のうち、ヌルでない最初の値を戻す。すべての引数がヌルである場合は、ヌルを戻す。引数を2つ以上指定する必要があります。
IFNULL(*arg1*, *arg2*)	○	*arg1*がヌルでない場合はarg1を、ヌルの場合は*arg2*を戻す。IFNULLは、基本的に、COALESCE()に引数を2つ指定するのと同じです。
NULLIF(*arg1*, *arg2*)	○	*arg1*と*arg2*が異なる場合は*arg1*を、等しい場合はヌルを戻す。データベースに含まれるヌルでない特定の値を、ヌルとして扱いたい場合に使用します。

SQL の集計関数

集計関数に 1 つの引数を渡す場合は、その前に DISTINCT というキーワードを挿入して、重複する値を省くことができます。

COUNT(*) を除くすべての集計関数では、ヌルと欠測値は無視されます。

関数	SQLite	説明
AVG(*num_expr*)		グループに含まれる行の、*num_expr* の列の平均値を計算する。*num_expr* の列は数値タイプでなければなりません。
COUNT(*expr*) COUNT(*)		グループの中で、*expr* の列がヌルでない行数を戻す。COUNT(*) は、グループに含まれる全行数を戻します。
GROUP_CONCAT(*expr*, *<separator = ','>*)	○	*expr* の列のうち、ヌルでない値をすべて連結して、文字列として戻す。*expr* の列が数値である場合は、文字に変換されます。区切り文字（*separator*）を指定した場合は、値の間に挿入されます。デフォルトの区切り文字はカンマです。DISTINCT は、GROUP_CONCAT() で *separator* が指定されていない場合のみ、使用できます。
MAX(*expr*)		グループの中で、*expr* の列の最大値を戻す（*expr* は、文字または数値）。
MIN(*expr*)		グループの中で、*expr* の列の最小値を戻す （*expr* は、文字または数値）。
STDDEV_POP(*num_expr*)		グループに含まれる行の、*num_expr* の列の母標準偏差を計算する。
STDDEV_SAMP(*num_expr*)		グループに含まれる行の、*num_expr* の列の標本標準偏差を計算する。
SUM(*num_expr*)		グループに含まれる行の、*num_expr* の列の合計値を戻す。すべてヌルの場合、SUM() はヌルを戻します。
TOTAL(*num_expr*)	○	SUM(*num_expr*) と同じだが、TOTAL() は、すべてヌルの場合、0.0 を戻す。
VAR_POP(*num_expr*)		グループに含まれる行の、*num_expr* の列の母分散を計算する。
VAR_SAMP(*num_expr*)		グループに含まれる行の、*num_expr* の列の標本分散を計算する。

索引
スクリプト構文リファレンス

記号

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z