```
--> /* ［数値処理］→［自動的に数値で出力］*/
    if numer#false then numer:false else numer:true;

--> /*浮動小数点の桁数=7に指定 */
    fpprec : 7;

--> /* 浮動小数点表示桁数を7に指定*/
    fpprintprec:7;

--> /* 2.1 基本統計量　*/

--> x:makelist(i, i, 1, 10);

--> /* 平均 */
    mean(x);

--> /* 分散 */
    x-mean(x);
    mean((x-mean(x))^2);
    mean(x^2)-mean(x)^2;
    var(x);

--> /* 標準偏差 */
    sqrt(var(x)); std(x);

--> /* 不偏分散*/
    var(x)*length(x)/(length(x)-1);
    var1(x);

--> /* 不偏分散の平方根 */
    sqrt(var1(x));
    std1(x);

--> kill(x);
    /* 変数 x クリア */

--> /* 大数の法則チェック */
```

```
--> /* 正規乱数を発生には, distrib パッケージが必要 */
    load(distrib);
    /* random_normal(m,s, n) */
    /* 平均m，標準偏差sの正規乱数をn個発生 */
    mean(random_normal(0,1,10));
    mean(random_normal(0,1,100));
    mean(random_normal(0,1,1000));
    mean(random_normal(0,1,10000));

--> /*　中心極限定理チェック */
    /* 2項乱数の発生には，distrib パッケージが必要 */
    load(distrib);
    /* random_binomial(n,p,x) */
    /* パラメータ(n,p)の2項乱数をx個発生 */
    S_n:random_binomial(1000000,0.5,10000);
    /* 標準化標本平均*/
    StdSampleMean: (S_n-1000000/2)/sqrt(1000000/4);
    load(descriptive); /* ヒストグラム描画にはdescriptiveパッケージが必要*/
    load(draw);
    draw2d(
    histogram_description(StdSampleMean, frequency=relative),
    /* histogram_description(データ,オプション)
    draw2d内ヒストグラム描画コマンド
    オプション
    nclasses=階級数, frequency=relative 相対度数表示 */
    explicit(pdf_normal(x,0,1),x, lmin(StdSampleMean),lmax(StdSampleMean)),
    /* pdf_normal(x,m,s) 平均m, 標準偏差sの正規分布密度関数
    limin(リスト)=リスト最小値，lmax(リスト)=リスト最大値*/
    yrange=[0,0.4]
    );

--> x:makelist(i, i, 1, 10);
    /* 最大値，最小値，メディアン，リスト要素和*/
    maxi(x); mini(x);median(x); lsum(i,i,x);

--> /* lsum(変数の関数, 変数, リスト) */
    /* 変数にリストの値を先頭から順に入れ、変数の関数値を計算し */
    /* それらの結果すべての和を返す */

--> /* pパーセンタイル */
    /* 記述統計パッケージ descriptive呼び込み*/
    load(descriptive);
    quantile(x,0); quantile(x,0.25); quantile(x,0.5);quantile(x,0.75);quantile(x,1);
```

```
--> /* 2.2 確率分布と乱数 */

--> /*メモリクリア*/
    kill(all);

--> /* 確率分布パッケージdistrib呼び込み*/
    load(distrib);

--> /* 標準正規分布密度関数グラフ */
    load(draw);
    draw2d(
    explicit(pdf_normal(x, 0,1), x, -4,4)
    );

--> /* 標準正規分布分布関数グラフ */
    draw2d(
    explicit(cdf_normal(x, 0,1), x, -4,4)
    );

--> /* 2.3.2 母平均のt検定*/
    kill(all);

--> /* t分布密度関数のグラフ*/
    load(distrib);
    load(draw);
    draw2d(
    key="n=100", color=red,
    explicit(pdf_student_t(x, 100), x, -4, 4),
    key="n=10", color=blue,
    explicit(pdf_student_t(x, 10), x, -4, 4),
    key="n=1", color=green,
    explicit(pdf_student_t(x, 1), x, -4, 4),
    ylabel="f(x)", xlabel="x"
    );

--> /*t分布下側2.5%点と上側2.5%点 */
    quantile_student_t(0.025, 10); quantile_student_t(0.975, 10);

--> /*t分布下側2.5%点分布関数値 */
    cdf_student_t(−2.228139, 10);
    cdf_student_t(quantile_student_t(0.025, 10), 10);
```

```
--> /*t分布上側2.5%点分布関数値 */
    cdf_student_t(2.2281389, 10);
    cdf_student_t(quantile_student_t(0.975, 10), 10);

--> /*t分布上側2.5%点上側確率 */
    1- cdf_student_t(2.2281389, 10);

--> data0: random_normal(0,1,20);
    data1:random_normal(1,1,20);

--> /*一標本 t 検定*/
    /* 標準化データ：平均=0の検定*/
    /*推測統計用パッケージstats呼び込み*/
    load(stats);
    /*一標本 t 検定 test_mean(データ)
    デフォルト信頼水準=95%*/
    test_mean(data0);

--> test_mean(data1);

--> /* 信頼水準99%の t 検定*/
    test_mean(data1, conflevel=0.99);

--> /* 信頼水準99%, 片側検定，対立仮説:平均<0 */
    test_mean(data1, conflevel=0.99, alternative=less);

--> /*二標本 t 検定*/

--> /* 平均の差の検定(等分散の場合)*/
    test_means_difference(data0,data1,varequal=true);

--> /* 平均の差の検定(母分散が異なる場合)*/
    test_means_difference(data0,data1);

--> kill(all);

--> /*2.4 回帰分析*/

--> /* データ入力 */
    stock:[4.8,-3.3,6.4,3,-0.4,2.6,-8.2,-2.5,6.9,-5.1,2.7,-7.9];
    eco1:[0.5,-1.8,5.5,-1.7,-0.9,2.9,-4.8,0.1,0.7,-2.6,1,-1.8];
    eco2:[0.1,-2.6,3.2,6.5,1.9,-2.4,-2.6,-1.5,5.3,0.9,-2.4,-0.6];
```

```
--> /* transpose(リスト) でリスト（列ベクトル）の転置 */
    /* addcol(行列，列ベクトル)で列ベクトルを追加した行列を作成 */
    /* データ行列の作成 */
    price:addcol(transpose(eco1), transpose(eco2),transpose(stock));

--> /* stock(=行列priceの３列目)の平均と不偏分散平方根*/
    load(descriptive) /* 記述統計パッケージ descriptive呼び込み*/;
    mean(col(price,3));
    std1(col(price,3));

--> /* eco1とstockの相関係数 */
    /*submatrix(行列, 列#) 行列から列#を除いた行列を返す*/
    /* cor(列ベクトル1, 列ベクトル2)
    列ベクトル1と列ベクトル2の相関行列を返す */
    cor(submatrix(price, 2));

--> /* eco2とstockの相関係数 */
    cor(submatrix(price, 1));

--> /* eco1とeco2の相関係数 */
    cor(submatrix(price, 3));

--> /* (eco1, eco2, stock)の相関行列 */
    cor(price);

--> /* (eco1, stock)散布図 */
    /* scatterplot([列データ１，列データ2], オプション)
    で(列データ1，列データ2)の散布図を描く.
    ただし，事前にdescriptiveパッケージの読み込みが必要*/
    scatterplot(
    submatrix(price, 2),
    point_size=3,
    /* 描画点の大きさをデフォルトの3倍に設定 */
    xrange=[-5,5], yrange=[-10,10],
    /* 横軸の範囲，縦軸の範囲を指定 */
    xlabel="Eco1", ylabel="Stock"
    /* 横軸ラベル，縦軸ラベルを指定 */
    );
```

```
--> /* stockをeco1に線形回帰 */
    /* statsパッケージ内
    linear_regression([x1列データ,...,xn列データ,y列データ])
    でｙの[x1,...,xn]への線形回帰を行う */
    load(stats);
    reg: linear_regression(submatrix(price, 2));

--> /* 線形回帰の結果
    回帰係数推定値
    b_estimation=[定数項，x1回帰係数,...,xn回帰係数]
    回帰結果の取り出しには，take_interfaceを用いる */
    a:take_inference(b_estimation,reg)[1];
    b:take_inference(b_estimation,reg)[2];

--> /* draw2dで散布図と回帰直線を描く*/
    /* 回帰直線の関数を作成 */
    f(x):=a+b*x;
    /* 散布図用eco1, stockデータ作成*/
    xy_data:makelist([eco1[i],stock[i]], i,1, length(eco1));

--> /* 散布図と回帰直線の描画 */
    load(draw);
    draw2d(
    point_size=3,
    xrange=[-5,5], yrange=[-10,10],
    points(xy_data), /* 散布図 */
    color=red,
    explicit(f(x),x,-5,5), /* 赤色，回帰直線 */
    xlabel="Eco1", ylabel="Stock"
    );

--> /* stockの[eco1, eco2]への多重線形回帰 */
    reg2: linear_regression(price);

--> /* 残差取り出し*/
    std1(take_inference(residuals, reg2));
```