0.6 ジュニア版　　数値計算

微分積分から始まって、非線形方程式の解、条件なし最適値問題、条件付最適値問題、そして線形計画法などを GAUSS で計算する方法を説明します。とりあえずは、そういった計算は必要ない方は、この章はスキップして章を読み進めてから戻ってきてください。

数値微分

f(x)=2Ｘ^3 について、 f'(1)を求める

プログラム

```
new;
cls;
fn f(x)=2*x^3;
x0=1;
print gradp(&f,x0);
```

画面表示

```
6.0000000
```

プログラム各行の説明

１行目　メモリを初期化する
２行目　スクリーンをクリアする
３行目　fn 命令を使って、f(x)=2Ｘ^{3}を定義する
４行目　１を変数 x0 に入れる（これを数値微分の評価点にする）
５行目　グラディエントを求める組込み関数 gradp を用いて、関数 f の x0 における数値微分を求めて、その結果を print 命令で画面表示する

注意事項

１）この方法は「数値微分」をしていて「代数的に解いた解に評価点の値を入れているわけではない」ので、一番上のケタを含めて５ケタ程度（例えば、答えが１であれば小数点以下３～４ケタまでしか信頼はない）例えば、x0=0;で０における数値微分をしてみればわかりますが、完全には０にはなりません
２）関数を別の関数で「インプットとして」呼び出すには＆のマークをつけて、＆ｆなどという具合に通常のインプットとは区別をする

上では、x0=1 に対する数値積分を解析的に条件を満たす値を探しにいっています。記号的に f'(x)=6Ｘ^{2}を先に求めてから、その微分した結果の関数に評価する点の値１を入れているわけではありません。したがって、場合によっては微妙にズレが生じます。しかしながら、その反対に記号的には容易には解けない「数値解析」の関数の扱い方を解説します。

数値積分

f(x)=2ｘ^3 について $\int_0^0 2x^3 dx$ を求める

プログラム

```
new;
cls;
fn f(x)=2*x^3;
x1={1,
    0};
print intsimp(&f,x1,1e-8);
print intquad1(&f,x1);
```

画面表示

```
 0.50000000
 0.50000000
```

プログラム各行の説明

１行目　メモリを初期化する

２行目　スクリーンをクリアする

３行目　fn 命令を使って、f(x)=2ｘ^3 を定義する

４～５行目　１と０を変数 x1 に列で入れる（これを積分微分の上限と下限にする）

６行目　組込み関数 intsimp を用いてシンプソン法で、上限と下限の間を 1e-8 の許容収束桁数で計算したものを print 命令で画面表示する

７行目　組込み関数 intquad1 を用いてガウス-ルジャンドル法で、上限と下限の間を計算したものを print 命令で画面表示する

注意事項

１）数値微分と同様に、一番大きなケタから数えて５ケタ程度の信頼しかない

２）積分の上限と下限は{上限,下限}の順である（数式に書いた上下関係どおりが採用されていて、上限と下限の順序を逆にすると、積分の性質により、値は負の値になる）

３）intquad2 および intquad3 を用いれば、同様にして２重３重までの数値積分が可能

上では、０から１までの f(x)=2ｘ^3 のｘに関する数値積分を２つの方法（シンプソン法とガウス-ルジャンドル法）によってそれぞれ求めています。数値微分の時と同様、先に関数を定義しておいて、それを数値積分を行なう組込み関数 intsimp または intquad1 に&をつけて関数のインプットであることを示して区別して入れます。同時に{上限,下限}の値を設定します。シンプソン法の法はさらに、インプットの第３要素として、収束計算の許容桁数を指定します（難しく考えないで、常に 1e-8 の値にを固定しても問題はありません）。

方程式の実数解

$x^5=1$ について、実数解を求める

プログラム

```
new;
cls;
fn f(x)=x^5-1;
start=1;
{x,retcode}=eqSolve(&f,start);
print x;
```

画面表示

```
1.0000000
```

プログラム各行の説明

1行目　メモリを初期化する

2行目　スクリーンをクリアする

3行目　fn 命令を使って、$f(x)=x^5-1$ を定義する（いま、$x^5=1$ を解こうとしている）

4行目　1 を変数 start に入れる（これを解をスタート値とする）

5行目　組込み関数 eqSolve を用いて関数 f をスタート値から収束するまで解を探し出す（その実数解を x、エラーコードを retcode とする）

6行目　x を画面表示する

注意事項

1）スタート値が、例えば0にすればリターンコードは1ではなくなり、その解は解ではない可能性がある（リターンコードが1のケースのみ解であると言える）

2）$ax^n+bx^{n-1}+\ldots=c$ の実数解が必要な場合、f(x)=a*x^n+b*x^(n-1)+…-c;というふうに右辺を反対側に以降したものを関数に定義する

3）print 命令以外の解を見つけ出す過程が示される（これを消して、リターンコードが1ではないケースにエラーメッセージを表示するプログラムは以下のようにします）

修正プログラム

```
new;
cls;
fn f(x)=x^5-1;
start=1;
screen off;
    {x,retcode}=eqSolve(&f,start);
screen on;
if retcode/=1;
    errorlog "ERROR: CANNOT FIND A ROOT.";
```

```
    else;
        print x;
    endif;
```

プログラム各行の説明

1行目　メモリを初期化する
2行目　スクリーンをクリアする
3行目　fn 命令を使って、f(x)=x^5-1 を定義する（いま、x^5=1 を解こうとしている）
4行目　1 を変数 start に入れる（これを解をスタート値とする）
5行目　スクリーンへの表示をオフにする
6行目　組込み関数 eqSolve を用いて関数 f をスタート値から収束するまで解を探し出す（その実数解を x、エラーコードを retcode とする）
7行目　スクリーンへの表示をオンにする（オフとオン間の結果は画面表示されない）
8行目　If 以下が真であれば（retcode が 1 ではなければ）次の命令を実行する
9行目　引用符内のエラーメッセージを画面表示する（errorlog は print 命令と同じ）
10 行目 If 以下が「そうでなければ」、次の命令を実行する
11 行目 x を画面表示する
12 行目 If 分岐の終りを示す

screen off;と screen on;により不要な途中経過を消せます。上のプログラムに start=0;を設定してみれば、エラーメッセージだけが表示されます。

注意事項

1）スタート値によって（たいていの場合 0 はいけない）解が見つからないケースもある
2）retcode が 1 ではないケースは（さまざまなケースがあるが）その答えは解ではない
3）print 命令と errorlog 命令は基本的に同じ（エラーメッセージには errorlog を使う）
4）この eqSolve は複数の本数の方程式を解けるほか、グローバル変数という下線のついた変数によって各種設定が可能（その際には、eqSolveSet;というグローバル変数の初期化を必要とする）

上の計算は、虚数解を除いて明らかに実数解が 1 であるケースについて扱いましたが、どんな複雑な関数でも「数値的に」この方程式をある条件で収束するまで動かすことで実数解を求めてきます。その際に、1 本の非線形方程式の場合、単に右辺の定数部分を左辺に移行して（右辺が 0 である場合には「＝0」の部分を取り除くだけ）関数を作っておいてそれを数値微分を求める gradp で、探し始める値とともに設定して、呼び出すだけです。この関数 gradp の場合、retcode が 1 ではない場合には、解を探し始めるスタート値が探し始めるにはふさわしくない場合が考えられます。その場合には、スタート値を何か他の値に変更してみてください。たいがいの場合、それだけでうまくいきます。

最適値計算(QNewton は最小値を求める)

max x^2-2x+2 ⇔ **min** $-(x^2-2x+2)$

プログラム

```
new;
cls;
qnewtonset;
fn fct(x)= -(x^2-2*x+2);
start=1;
{ x,f,g,ret}=QNewton(&fct,start);
print "x=" x;
```

画面表示

```
x=          1.0000000
```

プログラム各行の説明

１行目　メモリを初期化する

２行目　スクリーンをクリアする

３行目　QNewton のグローバル変数を初期化する（この場合なくても可）

４行目　fn 命令を使って、f(x)= $-(x^2-2x+2)$を定義する

５行目　1 を変数 start に入れる（これを解をスタート値とする）

６行目　関数の最小値を求める組込み関数 QNewton を用いて、関数 fct を呼び出して、スタート値 start=1 から最小値を探し出して、その結果をｘに入れる（この関数には解と最後のリターンコードを含めて合計４つのアウトプットリターンがあるが、このうち１つしか使わない）。

７行目　変数ｘの内容を、引用符内の説明メッセージとともに、画面表示する

この方法は、ニュートン法という最小値を求めるアルゴリズムに改良を加えた BFGS という方法を用いて数値解析によってその関数の最小値を求めるものです。最大値を求める場合には、関数全体にマイナスをつけて、マイナス関数の最小化を行なってください。ただし、確かにその点は微分が０になるような地点ではあるのですが、最大値や鞍点である可能性はあります。実際、上の fct(x)の関数の定義のマイナスの値を除いても、答えは同じ１になります。少し込み入ってきますが、関数に２つ以上のアウトプットリターンがある場合には、「変数＝関数」という形にはできなくて、「{変数 1,変数 1,…}＝関数」の形になります。上の場合は{x,f,g,ret}とマニュアルどおり４つの変数名をつけていますが（この場合、ｆはすでに使われているので、最小化する関数名自体にｆを使うことはできない）｛　｝の中に４つの変数がある限り、変数名は何であってもかまいません。{y,f,g,retcode}などと受けて、ｙを画面表示することもできるということです。

条件付最適値問題（sqpSolve は非線型線形の条件付最小値を求める）

max $x_1 x_2$　　s.t.　$2x_1+x_2=2$, $x_1 \geqq 0$, $x_2 \geqq 0$

⇔　**min** $-x_1 x_2$　　s.t.　$2x_1+x_2=2$, $0 \leqq x_1 \leqq 1e256$, $0 \leqq x_2 \leqq 1e256$

プログラム

```
new;
cls;
fn fct(x)= -x[1]*x[2];          /* Objective fuction to be minimized: min -x[1]*x[2]       */
sqpSolveSet;
_sqp_A={2 1};                   /*                  | x[1] |                                */
_sqp_b=2;                       /*   s.t. [ 2   1 ]|       | = 2 ,    x[1]>=0 and x[2]>=0.   */
_sqp_Bounds={0 1e256,           /*                  | x[2] |                                */
                0 1e256};
start={0,0};
screen off;
    {x,f,lagr,retcode}=sqpSolve(&fct,start);
screen on;
x=round(10000*x)/10000;         /* Round below 0.0001 */
print/rz "x=" x;
```

画面表示

```
x=
                0.5
                  1
```

プログラム中のパラメータの設定

目的関数

　fn fct(x)=関数名(x[1]と x[2]からなる)

等号条件

　_sqp_A*x　＝　_sqp_B

不等号条件

　_sqp_C*x　≧　_sqp_D

境界条件

　_sqp_Bounds={　第１変数の下限　第１変数の上限,　　←１変数ごと１行

　　第２変数の下限　第２変数の上限　**}**　　←１変数ごと１行

（上の場合、ｘは x[1]と x[2]からなる２×１のベクトル。（１×２）×（２×１）＝１×１であることに注意。ここでは、不等号条件はないので_sqp_C と_sqp_D とは省略。）

境界条件のところは、０以上の場合、上限は 1e256 と莫大な桁数の数を割り当てて事実上ないものとすることが GAUSS では常道になっています（下限は-1e256 となる）。

線形計画法

$\max\ 3x_1+4x_2+2x_3 \quad \Leftrightarrow \quad \min\ -(3x_1+4x_2+2x_3)$

$3x_1+2x_2+4x_3 \leqq 15 \qquad -3x_1-2x_2-4x_3 \geqq -15$

$x_1+2x_2+3x_3 \leqq 7 \qquad -x_1-2x_2-3x_3 \geqq -7$

$2x_1+x_2+x_3 \leqq 6 \qquad -2x_1-x_2-x_3 \geqq -6$

$x_1 \geqq 0,\ x_2 \geqq 0,\ x_3 \geqq 0 \quad 0 \leqq x_1 \leqq 1e256,\ 0 \leqq x_2 \leqq 1e256,\ 0 \leqq x_3 \leqq 1e256$

プログラム

```
new;
cls;
fn fct(x)= -(3*x[1]+4*x[2]+2*x[3]);   /* Objective function to be minimized */
sqpSolveSet;
_sqp_C={-3 -2 -4,                      /* C*x>=d multiplying by -1. */
        -1 -2 -3,
        -2 -1 -1};
_sqp_d={-15,
        -7,
        -6};
_sqp_Bounds={0 1e256,                  /* x[1]>=0, x[2]>=0, x[3]>=0 */
             0 1e256,
             0 1e256};
start={1,1,1};
{x,f,lagr,retcode}=sqpSolve(&fct,start);
print/rz "x=" x;
```

画面表示

```
x=
      1.6666667
      2.6666667
              0
```

上の場合は、（非線型の）条件付最適化の等号条件は使わなくて、今度は不等号条件だけを使ったものです。なお、**_sqp_C** と**_sqp_D** は

```
_sqp_C *x >= _sqp_D
```

というように左辺が大きくなるのように設定しますので、符号を逆転させるには、すべての両辺の係数に－１をかけます。前の例と基本的に同じですが、ここでは３変数なので、境界条件も３行分必要となります。当然ながら、スタートベクトルも３×１になります。

２次計画問題（Quadratic Program）

マコービッツモデルなどで分散共分散行列を用いて、そのウエイト計算をする際に使われるのが、２次が最高次の条件付最小化を行なうこの Quadratic Program です。形式は、

$$\min \frac{1}{2} x'Qx - x'R$$
$$s.t. Ax = b$$
$$Cx \geq d$$

となります。行列とベクトルで表されているの少し複雑に感じますが、通常はｘ'R の項はなくて、等号制約だけであるとするならば、σを分散共分散、ｗをウエイトとして

$$\min \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij}$$
$$s.t. \quad \sum_{i=1}^{n} w_i = 1$$

を満たすｗのベクトルを求めます。上の GAUSS での設定で言うならば、Q には分散共分散行列がきて、その両側からｗのベクトルをかけます。これによって行ｉと列ｊの自己同士を含むすべての組み合わせの２次項ができます。二分の１は計算上の都合からついています。３つのウエイトからなる場合、分散共分散行列は３×３のディメンションになります。等号制約のところは、A を１×３のすべてが１からなる行ベクトルとして、今求める３×１のウエイトベクトルとかけ合わせると、ちょうどウエイトの合計ができます。GAUSS では大文字も小文字も区別がありませんから、

Q＝分散共分散行列
A={ 1 1 1 }
b=1

としてやって、上のような２次形式の条件付最小化がなされるような３×１のｘベクトル（ここではｗのこと）を求めます。その際、R は求めるｘベクトルと同じディメンションの零ベクトルである必要があります。不等号制約の係数および境界条件の係数は使わないので、すべてを０とします。いま、簡単化のため、分散共分散行列を対角成分がすべて１でその他の成分が０の単位行列とします（分散がすべて１、共分散がすべて０）。

プログラム

```
new;
cls;
Q={1 0 0,
   0 1 0,
   0 0 1};
R={0,0,0};
A={1 1 1};
```

```
b=1;
C=0;
d=0;
bnds=0;
start={1,1,1};
{ x,u1,u2,u3,u4,ret } = QProg( start,q,r,a,b,c,d,bnds );
print x;
```

画面表示

```
0.33333333
0.33333333
0.33333333
```

上のように、求めるｘベクトルと同じディメンション（同じ個数の成分）からなるスタートベクトルを適当に決めてやって、Qprog を呼び出して２次問題を解かせます。上のアウトプットリターンのうち、最初のｘに相当する部分が解のベクトルになります。解に境界条件がある場合には、上の bnds のところを非線形線形の計画問題と同じようにパラメータ設定します。

練習問題

【問１】

$2x^3+3x^2+5x=1$ の実数解を求めよう。

【問２】

$3x^2-2x+5$ の最小値を求めよう。

【問３】

非負の領域において、予算制約 $3x_1+5x_2=150$ のもとで、効用 $U(x_1,x_2)=x_1x_2$ を最大化する点(x_1,x_2)を求めよう。

発展

⇒　本編　4.1　4.2

⇒　Luenberger　2.6c　5.2a　6.6a　6.7a