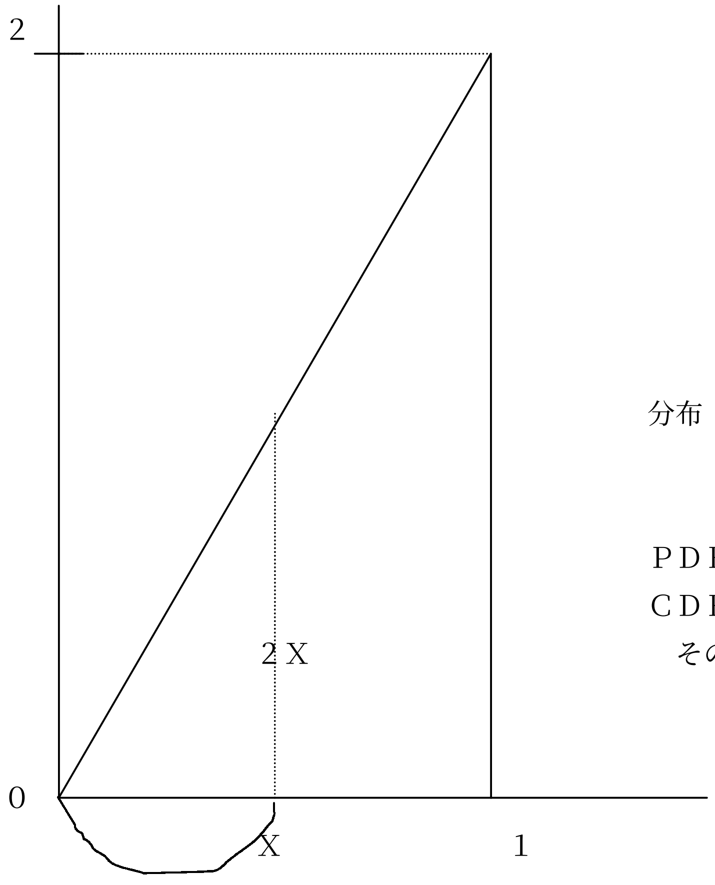
5.18 モンテカルロシミュレーション 逆関数法 ver.0.1

ここでは、少し初歩に戻って、０と１の間の一様乱数を用いて、それぞれの分布の逆関数にこれをあてはめる形で乱数を生成する方法を体系的に学んでおきましょう。様々な分布にしたがう乱数を擬似的に生成するにはいくつかの方法がありますが、その中で最も簡便なのが、この逆関数法と呼ばれるものです。分布には、ほとんどの場合、明示的な形で PDF と CDF があります。そのなかの分布関数と呼ばれる CDF（大文字の F(x)で表される）を、一様乱数の０と１の変域に対して、逆関数を計算したもの $F^{-1}(x)$に０と１の一様乱数を代入して乱数を発生させます。逆関数 $F^{-1}(x)$が明示的に解くことができる場合には、この方法は極めて簡便かつ強力です。最近の行列系言語では、このやり方はスピードを気にせずとも瞬時になされます。統計学的性質としては、この逆関数の変換は単調性と存在するのであれば相関性を保存します。このことは、分散減少やオーダーに関する性質をそのまま変換後の分布にも適応できるということです。ただし、すべての分布が明示的に $F^{-1}(x)$の形で解くことができるわけではありません。その場合には、その他の乱数生成法を用いるか、もしくは逆関数 $F^{-1}(x)$の近似値を計算するアルゴリズムに代替する必要があります。

<u>Triangular Distribution</u>



分布：大きな三角形
面積　１

ＰＤＦ：２ｘ　線分
ＣＤＦ：ｘ²　小三角形
その Range: 0≦ｘ²≦１

必ずしも上のような向きや形の三角形である必要はないのですが、大きな全体の三角形の面積が１になるように三角関数の PDF を設定してやると、それぞれの位置ｘに対する小さな三角形の高さ２ｘの線分を－∞からｘまで積分すると（すなわち、この場合０とｘの間のｘにしか実質上分布は存在しないから）０からｘまでこの線分２ｘを動かした面積 1/2×ｘ×２ｘ＝x^2が CDF となります。この CDF は、ｘが０から１に動くにつれて、０から最大１まで動きます。このように CDF というのはどんな分布でも０から１までの range を持つように定義されています。つまり、小さな三角形x^2の取り得る値は０と１の間ということになります。ここで、０と１の間の一様関数を思い出してください。その場合、ｘの取り得る domain は０から１でした。もし、小さな三角形（ＣＤＦ）の逆関数を計算して、その range と domain を逆にすることによって、この一様関数をそこに均等に流し込むことができれば、この三角分布の乱数も作成できるはずです。そして、できあがった三角分布の乱数の頻度をグラフにすると、ほぼ大きな三角形の形と一致するはずです。

　この三角分布の場合、ＣＤＦはx^2ですから、$y=x^2$のｙとｘを交換して、

$$x = y^2 \Rightarrow y = \sqrt{x} \quad 0 < x < 1$$

というような逆関数が簡単に計算できるわけです。このルートｘのところに０から１までの一様乱数を流し込みます。下のプログラムでは、ｒ列ｃ行の三角分布乱数を作成する関数を考えるにあたって、まず一様乱数を同じくｒ行ｃ列分だけ先に発生させてｘに代入しておきます。これを今の CDF の逆関数ルートｘに代入して、それをリターンとして返しただけの簡単なものです。これを１万行１列分（つまり１万個）生成して、19 個の棒グラフのヒストグラムにして表示させています。

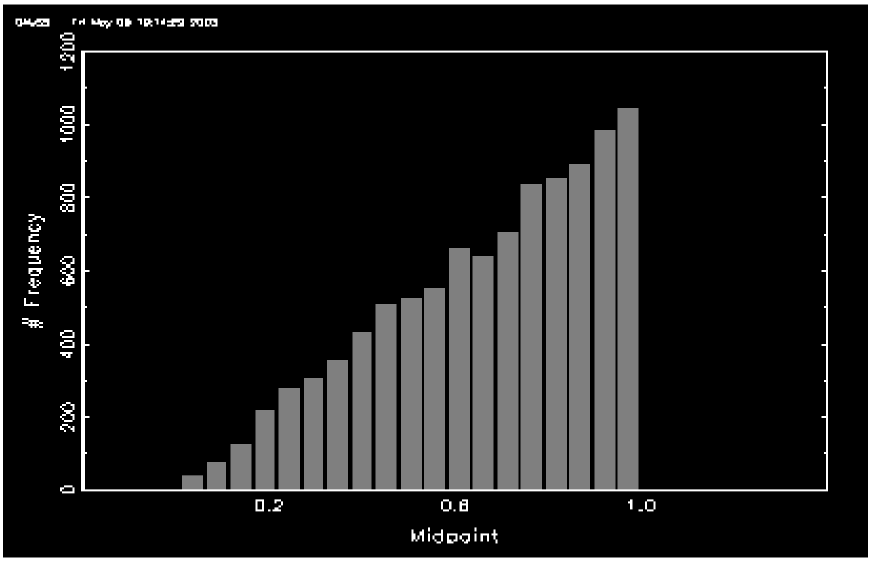
プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndtri(10000,1),19);

proc rndtri(r,c);
    local x;
    x=rndu(r,c);
    retp( sqrt(x) );
endp;

proc rndtri2(r,c);
    local x;
    x=rndu(r,c);
```

```
    retp( 1-sqrt(x) );
endp;
```

次のグラフは関数 rndtri を用いたものですが、１からそのリターンを引いたものを関数にした rndtri2 を用いれば、グラフは反転して０の方が頻度が大きくて減少していくものになる。各自試してほしい。その他の形の三角分布についても、全体の大きな三角形の面積が１でありさえすれば、簡単に計算して乱数化することができる。

グラフ表示



<u>Exponential Distribution</u>

　次に、一般の分析で使われる緒分布について、CDF の逆行列がすぐ計算できるものを示しておこう。これらは、すべて０から１の間のものに変換されていることに注意しよう。まさに、それが CDF の定義であるからだ。大文字の F(x)を CDF であるとすると、

$F(x)=1-\exp(-\lambda x) \qquad \lambda>0 \quad$ ただし $x \geqq 0$

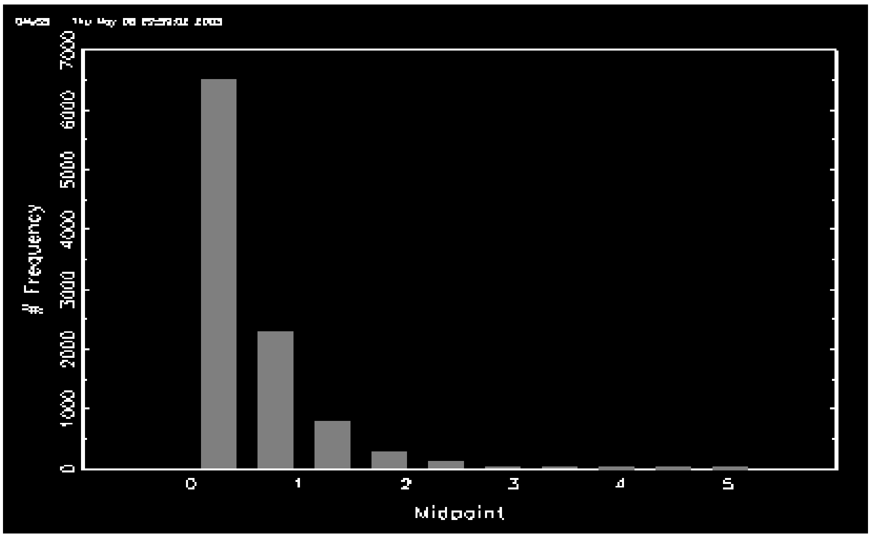
$F^{-1}(x)=-\ln(1-x)/\lambda \qquad \lambda>0 \qquad 0<x<1$

というふうに計算できる。$y=1-\exp(-\lambda x)$ の y と x を交換して、$x=1-\exp(-\lambda y)$。そして、$1-x=\exp(-\lambda y)$ の両辺の自然対数をとると、$\ln(1-x)=-\lambda y$ となる。y イコールの式になおすと、結局、$y=-\ln(1-x)/\lambda$ となる。この x に０と１の間の一様乱数を代入すれば、この λ というパラメータに対する指数分布の乱数が完成する。プログラムに関しては、λ が０または負の値に対してエラーメッセージを出すこと以外はほとんど同じ要領である。まず、一様乱数を発生させ x に代入し、逆関数 $-\ln(1-x)/\lambda$ をリターンとして返すだけである。これを、ヒストグラムで適当なバーの数にしてグラフ表示させているのである。ちなみに、指数関数に対する PDF は、$f(x)=\lambda\exp(-\lambda x)$ となる。ヒストグラムは、x を０から∞まで動かした時のこの f(x)の軌跡と形がほぼ一致するはずである。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndexp(10000,1,2),10);

proc rndexp(r,c,lambda);
    local x;
    if lambda<=0;
        errorlog "ERROR: Parameter lambda must be a positive number.";
        retp(".");
    endif;
    x=rndu(r,c);
    retp( -ln(1-x)/lambda );
endp;
```

グラフ表示



Cauchy Distribution

この関数に対する PDF、CDF それにその逆関数は次のように書ける。

$$f(x)=1/(\pi(1+x^2))$$

$$F(x)=1/2+1/\pi\tan^{-1}(x)$$

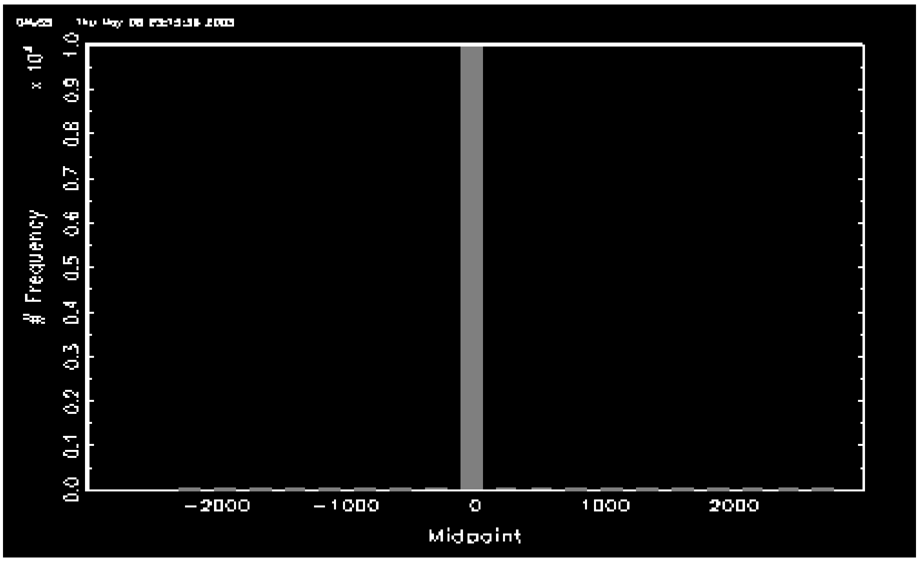
$$F^{-1}(x)=\tan(\pi(x-1/2))$$

なお、下のプログラムには関係ないが $\tan^{-1}(x)$は GAUSS では arctan(x)で実現されます。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndcauchy(10000,1),19);

proc rndcauchy(r,c);
    local x;
    x=rndu(r,c);
    retp( tan(pi*(x-1/2)) );
endp;
```

グラフ表示



なお、上の例は、パラメータのない Cauchy 分布のケースであるが、a と b のパラメータがあるケースの PDF、CDF それにその逆関数は次のようになる。

$$f(x)=1/(\pi b(1+((x-a)/b)^2))$$

$$F(x)=1/2+1/\pi \tan^{-1}((x-a)/b)$$

$$F^{-1}(x)=a+b\tan(\pi(x-1/2))$$

上のグラフの例は、この a＝0 かつ b＝1 のケースにあたる。rndcauchy(r,c,a,b)の形に proc を書き換えて、自分でプログラムを書きなおしてみよう。

Pareto Distribution

パレート分布に対する CDF とその逆関数は次のようになる。

$F(x)=1-(b/x)^a \qquad x \geqq b$

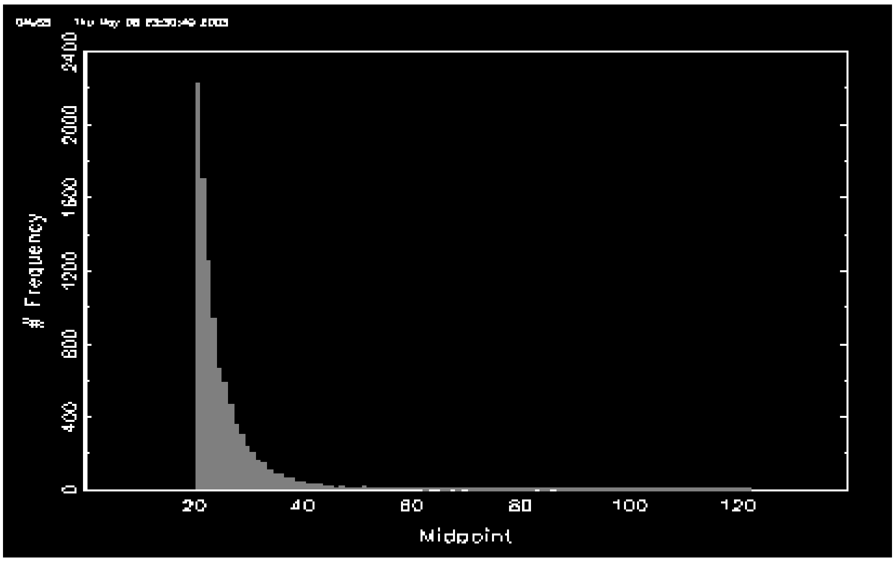
$F^{-1}(x)=b/((1-x)^{1/a})$

まったく、同様にして逆関数に０と１の間の一様乱数を代入すればよい。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndpareto(10000,1,5,20),100);

proc rndpareto(r,c,a,b);
    local x;
    x=rndu(r,c);
    retp( b/(1-x)^(1/a) );
endp;
```

グラフ表示



Logistic Distribution

すべての範囲のｘを取り得るロジスティック分布の CDF と逆関数は次のようになる。

$F(x)=1/(1+\exp(-x))$

$F^{-1}(x)=\ln(x/(1-x))$

となるから、プログラムではｘを１－ｘで割る時に要素対要素の計算にすることに気をつけてドットをつけて、同様に０と１の間の一様乱数を代入する。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndlog(10000,1),19);

proc rndlog(r,c);
    local x;
    x=rndu(r,c);
    retp( ln(x./(1-x)) );
endp;
```

Weibull Distribution

０または正の値のｘに対するワイブル分布の CDF とその逆関数は次のようになる。

$F(x)=1-\exp(-(x/\alpha)^{\gamma})$　　$\alpha,\gamma>0$　　ただし $x\geqq 0$

$F^{-1}(x)=\alpha(-\ln(1-x))^{1/\gamma}$

となるから、αとγに関して正でない値に対してエラーメッセージがでるようにプログラムを書き加えて、同様に０と１の間の一様乱数を逆関数に代入する。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndweibull(10000,1,2,10),19);

proc rndweibull(r,c,alpha,gam);
    local x;
    if alpha<=0;
        errorlog "ERROR: Parameter alpha must be a positive number.";
        retp(".");
    endif;
    if gam<=0;
        errorlog "ERROR: Parameter gamma must be a positive number.";
        retp(".");
    endif;
    x=rndu(r,c);
    retp( alpha*(-ln(1-x))^(1/gam) );
```

```
endp;
```

Gumbel Distribution

　パラメータなしの単純なタイプのガンベル分布の CDF とその逆関数は次のようになる。

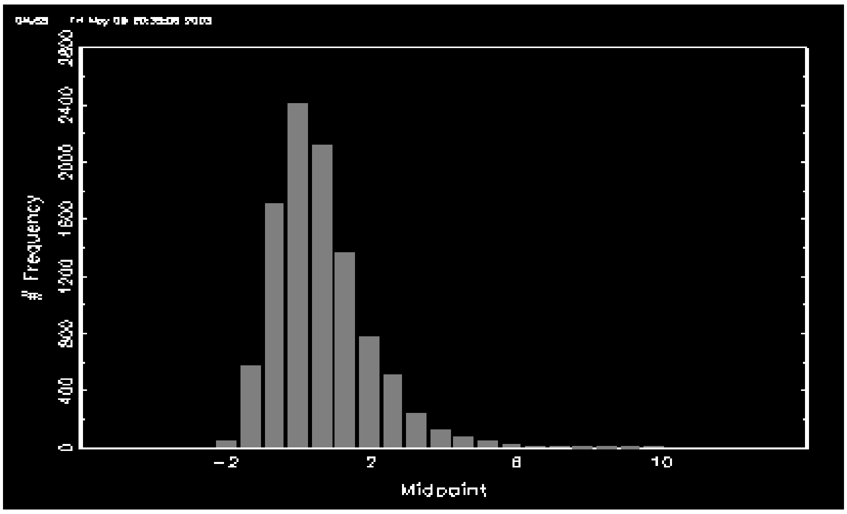
$$F(x)=\exp(-\exp(-x))$$

$$F^{-1}(x)=-\ln(-\ln(x))$$

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndgumbel(10000,1),19);

proc rndgumbel(r,c);
   local x;
   x=rndu(r,c);
   retp( -ln(-ln(x)) );
endp;
```

グラフ表示



なお、パラメータ a とｂの２つがあるケースでは、

$$F(x)=\exp(-\exp(-(x-a)/b))$$

$$F^{-1}(x)=a-b\ln(-\ln(x))$$

ちょうど上のグラフのケースは、a＝0 かつ b=1 のケ－スにあたる。rndgumbel(r,c,a,b)とすることで、簡単に２パラメータのケースに変更できるので自分で変更してみよう。
これまでの分布は、CDF に対して明示的な形で逆関数が示せるものを扱いました。その他 Inverse CDF を近似によって計算して、これまでのやり方と同じように０と１の間の一様

乱数を代入して求める分布としては次のようなものがあります。

Student’s ｔ Distribution

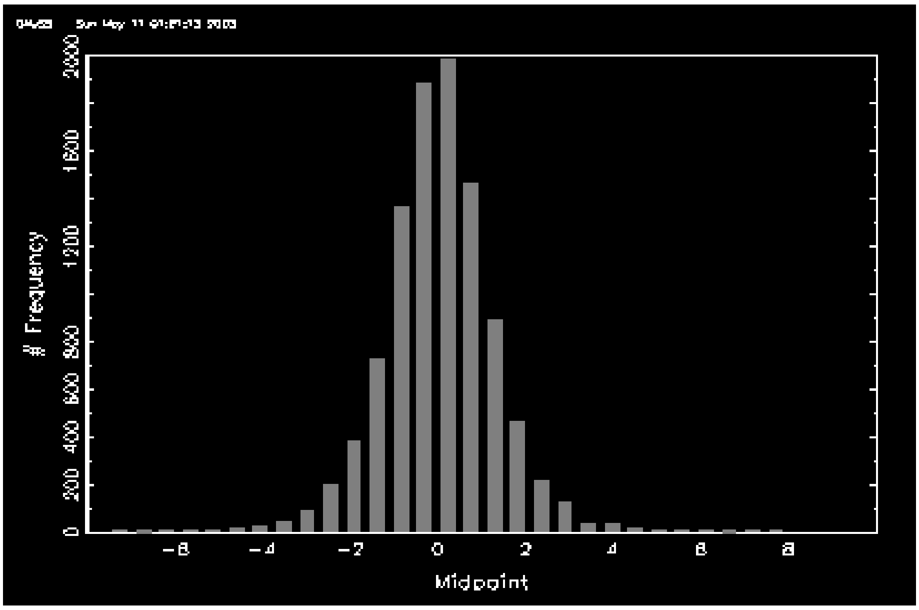
　基本的にほとんどの分布の逆関数が近似で求められるのですが、ごく簡単な例としては、Student t があります。これらの分布の逆関数の compliment を求める関数が GAUSS には標準で装備されていますので、代入すべき０と１の間の一様乱数を１から差し引きすることによって求めます（ただし、これは乱数なので実質上１から差し引きをする必要はありませんが）。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndt(10000,1,5),29);

proc rndt(r,c,df);
    local x;
    if df<=0;
        errorlog "ERROR: Parameter df must be a positive integer.";
    retp(".");
    x=rndu(r,c);
    retp( cdftci(1-x,df) );
endp;
```

下のグラフは df＝5 のケースを示しています。ただし、この逆関数の compliment を求める関数は標準で装備はされていますが、あくまでも近似なので、これまでのようなきれいに逆関数が解けるケースとは分けて考える必要があります。

グラフ表示



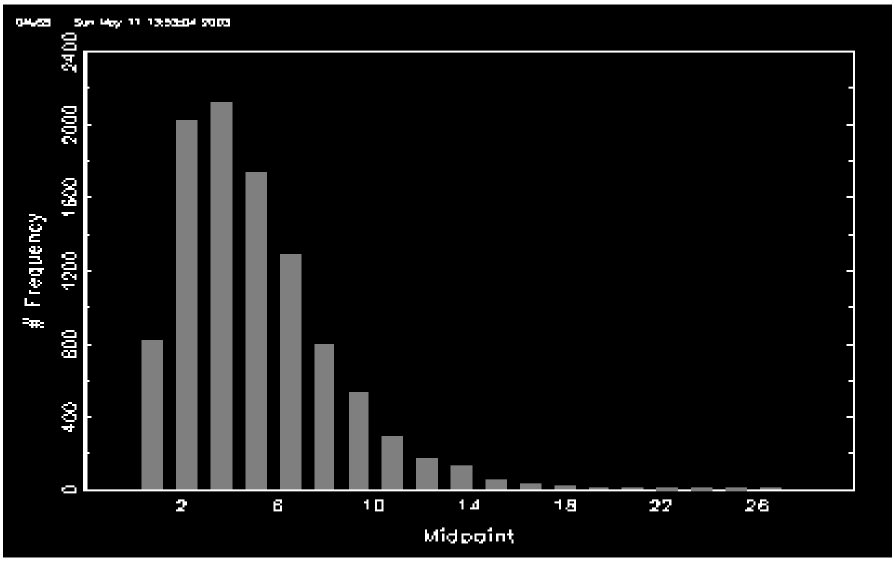
<u>χ^2 Distribution</u>

　その他、GAUSS の標準関数を使って簡単に逆関数が作れて、それを用いて乱数を発生できるものとして χ^2 分布があります。この場合の CDF の逆関数を求める１つの方法として簡便な方法として、間接的にガンマ関数の inverse incomplete を用いる方法があります。

プログラム

```
new; cls;
library pgraph;
graphset;
call hist(rndx2(10000,1,5),19);

proc rndx2(r,c,df);
    local x;
    if df<=0;
        errorlog "ERROR: Parameter df must be a positive integer.";
    retp(".");
    endif;
    x=rndu(r,c);
    retp( gammaii(df/2,x)*2 );
endp;
```

グラフ表示



ただし、上のやり方は若干間接的であって、GAUSS にはガンマ分布にしたがう乱数を発生させる関数が標準でありますから、それを用いて同じようにして下のようにして乱数を発生させることの方がより現実的です。下のこちらのケースは逆関数法ではありません。

プログラム（直接ガンマ乱数発生関数を利用）

```
proc rndchis(r,c,df);
    local x;
    if df<=0;
        errorlog "ERROR: Parameter df must be a positive integer.";
    retp(".");
    endif;
    x=rndu(r,c);
    retp( rndgam(r,c,df/2)*2 );
endp;
```

<u>Geometric Distribution</u>

その他、逆関数を自分でプログラムしなければならない例として、離散系の幾何分布があります。

$P(X = i) = pq^{i-1}$, 　$i=1,2,3,4,\ldots.$ 　ここで、$q=1-p$

$F(i) = 1 - P$(最初の i 回の試行が失敗)$= 1-q^{i}$

いま、$F(i-1) \leqq X < F(i)$ならば、$X = i$

これは、$1-q^{i-1} \leqq X < 1-q^{i}$ すなわち $q^{i} < 1-X \leqq q^{i-1}$ ならば $X = i$

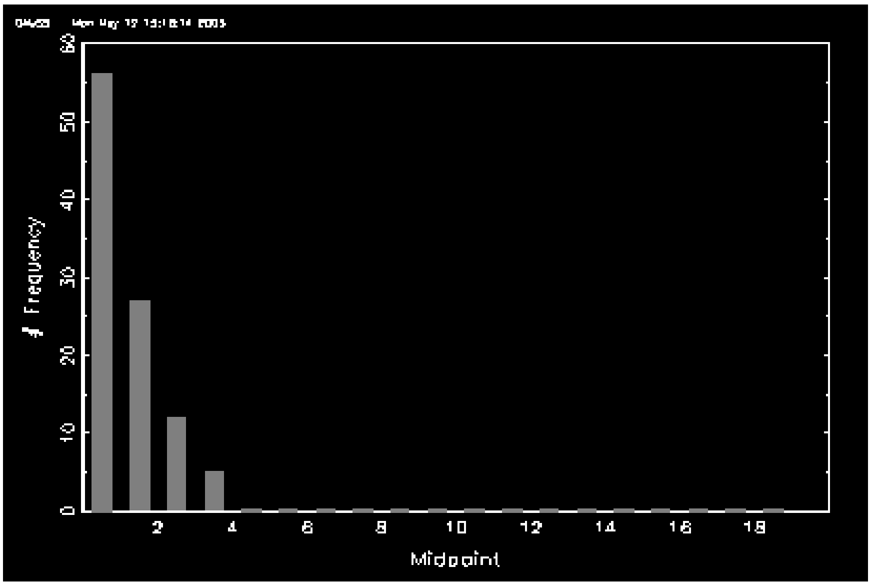
$X = \min[\, i \mid q^{i} < 1-X] = \min[\, i \mid i \ln(q) < \ln(1-X)] = \min[\, i \mid i > \ln(1-X)/\ln(q)]$

$= \mathrm{ceil}[\ln(1-X)/\ln(1-p)]$ 　このXのところに０と１の間の一様乱数を代入します。

プログラム

```
new; cls;
library pgraph;
graphset;
v=seqa(0,1,20);                    /* break points */
call hist(rndgeo(100,1,0.6),v);

proc rndgeo(r,c,p);
    local x;
    if p<=0 or p>=1;
        errorlog "ERROR: Prob p must be o<p<1.";
        retp(".");
    endif;
    x=rndu(r,c);
    retp( ceil(ln(1-x)/ln(1-p)) );
endp;
```

グラフ表示



上のように、頻度は 1,2,3,..という値しかとりません。ヒストグラムを表すのに、区切りの数を第２要素に設定するのではなくて、Break Points のベクトル v を設定しています。０以下はないのですが、グラフが乱れないように０から 0,1,2,3,..,19 というふうに設定してあります。それ以下のものの区間にくる頻度を数えてプロットしたものです。

　この他、inverse CDF の関数がわかりさえすれば、一様乱数を代入して、理論上どのような分布の乱数も作ることが可能です。公開されているライブラリなどをもとに(GAUSS

上のライブラリは初歩的な誤りが多いので自分で確認が必要)複雑なプログラムを必要とする inverse CDF などのケースについても、すべてもともとの CDF の range は０と１の間にありますから、inverse CDF の domain は０と１の間になり、一様乱数を代入してその乱数を生成できるわけです。

　一般に、逆関数法はスピードの面で劣るという記述がありますが、この種の記述は行列系の言語の登場以降は正しい記述とは言えないものとなっています。また、乱数を 1−X に代入するのと X 自身に代入するのは同じことで、X 自身に代入した方が効率的であるという記述も多々見られますが、これも現在では正しい記述とは言えなくなってきています。このように、GAUSS 自身にそれほど多くの乱数関数が装備されていないくても、プログラムを書く側にとってはほとんど何の問題もないことが実感できたと思います。また、仮に多くの種類の乱数がその言語に標準装備されていたとしても、どういうアルゴリズムで書かれているかが重要です。この章の前半で扱ったきれいに明示的に逆関数が解ける場合には、何ら問題はなくこの方法は非常に簡便でかつ強力なのですが、複雑な計算や他の分布の関数を間接的に使って計算される逆関数に一様乱数を代入する場合には、特に注意が必要です。グローバルにその乱数が有効であるかどうか各自確かめて見る必要があります。間接的に使われる関数やプログラム上の近似される箇所が多いほど、その乱数過程は信頼度の低いものになります。その場合には、定義自身にもどってシンプルに分布を合成して乱数を発生させるとか、逆関数法以外のアルゴリズムを使うことが考えられます。