NEW SPECIAL FUNCTIONS IN GAMS

ERWIN KALVELAGEN

Abstract. This document describes the new gamma and beta functions in GAMS.

1. Introduction

In response to requests by users we have added a number of special functions to GAMS. This document describes their use and their implementation.

function	**(D)NLP**	**description**	**domain**
`gamma(x)`	DNLP	gamma function $\Gamma(x)$	$x \neq 0, -1, -2, ...$
`loggamma(x)`	NLP	$\ln \Gamma(x)$	$x > 0$
`gammareg(x,a)`	NLP	incomplete gamma function $\gamma(x, a)$	$x \geq 0, a > 0$ $x > 0$ for derivatives
`beta(x,y)`	DNLP	beta function $B(x, y)$	$x, y, x + y \neq 0, -1, -2, ...$
`logbeta(x,y)`	NLP	$\ln B(x, y)$	$x, y > 0$
`betareg(x,a,b)`	NLP	incomplete beta function $I_x(a, b)$	$0 \leq x \leq 1, a, b > 0$ $0 < x < 1$ for derivatives
`binomial(x,y)`	NLP	generalized binomial coefficient $\binom{x}{y}$	$x, y \neq -1, -2, ...$

Table 1. Special functions

GAMS	**Mathematica**	**Matlab**	**Numerical Recipes**
`gamma(x)`	`Gamma[x]`	`gamma(a)`	`exp(gammln(x))`
`loggamma(x)`	`LogGamma[x]`	`gammaln(x)`	`gammln(x)`
`gammareg(x,a)`	`GammaRegularized[a,0,x]`	`gammainc(x,a)`	`gammp(a,x)`
`beta(x,y)`	`Beta[x,y]`	`beta(x,y)`	`beta(x,y)`
`logbeta(x,y)`	`Log[Beta[x,y]]`	`betaln(x,y)`	`log(beta(x,y))`
`betareg(x,a,b)`	`BetaRegularized[x,a,b]`	`betainc(x,a,b)`	`betai(a,b,x)`
`binomial(x,y)`	`Binomial[x,y]`	`nchoosek(n,k)`	`bico(n,k)`

Table 2. Comparison of special functions

Table 1 summarizes the new special functions available in GAMS. We mention the equivalent routines in Mathematica, Matlab and Numerical Recipes, Chapter 6[34] in table 2. Note that we only consider real arguments and results, while the Mathematica functions are defined in terms of the Complex plane.

Date: April, 2004, updated July, 2004; Nov, 2004; March, 2005; October, 2006.

It is noted that we don't provide the inverse forms of these functions. If you need to find say x with

$$x = \Gamma^{-1}\left(\sum_i y_i\right) \tag{1}$$

then we can formulate the equation

$$\sum_i y_i = \Gamma(x) \tag{2}$$

or

```
x.lo = 0.001; x.up = 50;
gammadef.. sum(i, y(i)) =e= gamma(x);
```

which lets the NLP solver solve this equation for x.

2. Old stuff: the error function

The error function is often defined by[2, 45, 42]:

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{3}$$

The name "error function" is coined by [21] to indicate its connection with probability theory. Originally the notation Erf(.) was used, which later became erf(.) [29].

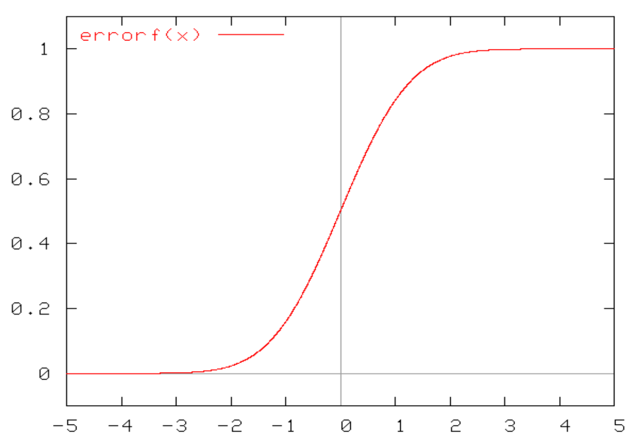

Figure 1. The errorf(x) function

The function `errorf(.)` in GAMS implements a variant on this:

$$\begin{aligned} \mathrm{errorf}(x) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{1}{2}t^2} dt \\ &= \frac{1 + \mathrm{erf}(x/\sqrt{2})}{2} \end{aligned} \tag{4}$$

which is the cumulative distribution function of the standard normal distribution $N(0,1)$. See figure 1 for a graph of this function. Some relevant values are:

$$\begin{aligned} \text{errorf}(0) &= \frac{1}{2} \\ \lim_{x\to-\infty} \text{errorf}(x) &= 0 \\ \lim_{x\to\infty} \text{errorf}(x) &= 1 \end{aligned} \tag{5}$$

It is known from statistics that if $X \sim N(\mu, \sigma^2)$ then

$$\frac{X-\mu}{\sigma} \sim N(0,1) \tag{6}$$

I.e. we can use `errorf((X-mu)/sigma)` to express a Normal distribution function with mean μ and variance σ^2.

A related distribution is the lognormal distribution. A stochastic variable $X > 0$ has a lognormal distribution if $Y = \ln(X)$ is normally distributed. More precisely, when we introduce a location parameter μ and a scale parameter $\sigma > 0$ then the distribution function is

$$\text{errorf}\left(\frac{\ln(X)-\mu}{\sigma}\right) \tag{7}$$

3. The Gamma function

The gamma function [2, 24, 46, 42, 47] is defined by Euler's integral

$$\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt \tag{8}$$

The function is related to the factorial function as follows:

$$\Gamma(n) = (n-1)! \tag{9}$$

for integer arguments n.

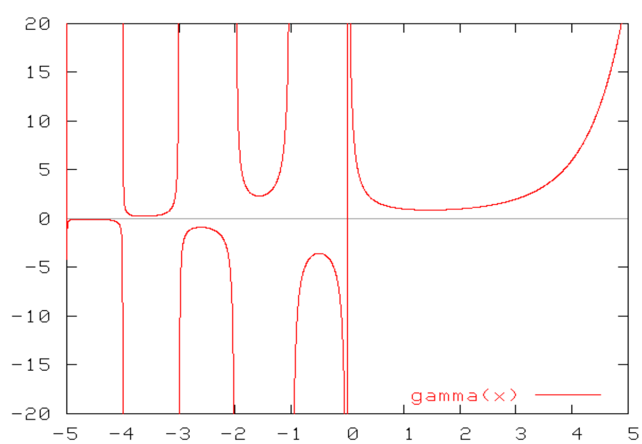

Figure 2. The `gamma(x)` function

The gamma function is available under GAMS as `gamma(.)`. It is based on an implementation of Cody[10] available from Netlib[9].

The gamma function becomes large very quickly: $\Gamma(15) > 10^{10}$, $\Gamma(72) > 10^{100}$ and $\Gamma(451) > 10^{1000}$. GAMS will trigger a domain error as soon as $x > 70$. The gamma function is not defined for the integer values $x = 0, -1, -2, \ldots$. For these values also a domain error is triggered. Safe bounds for being able to call `gamma(x)` (and its derivatives) are `x.lo=1.0e-5` and `x.up=69.0`.

For larger arguments, we supply the `loggamma(x)` function, which requires an argument $x > 0$. It returns the (natural) logarithm of the gamma function $\ln \Gamma(x)$. Note that it is not advised to form the equation

```
x.lo = 0.001;
eq.. y =e= exp(loggamma(x));
```

but rather

```
x.lo = 0.001;
y.lo = 0.001;
eq.. ln(y) =e= loggamma(x);
```

which can be considered as applying a non-linear scaling on $y = \Gamma(x)$.

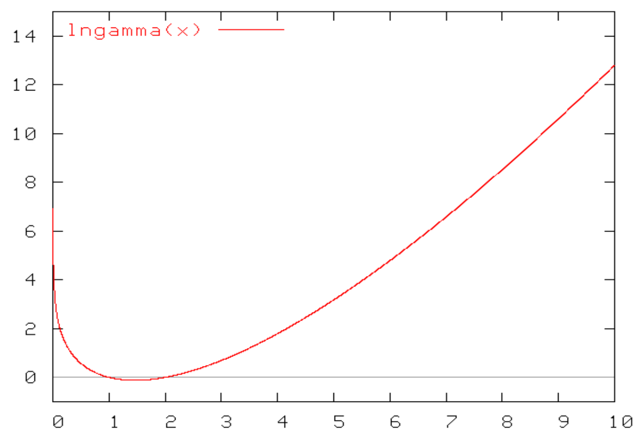

FIGURE 3. The `loggamma(x)` function

Both the `gamma(.)` and `loggamma(.)` can be used inside model equations. When called by an NLP solver both first and second derivatives can be provided.

The derivative of the gamma function is implemented by evaluating the expression

$$\frac{d\Gamma(x)}{dx} = \Gamma(x)\Psi(x) \tag{10}$$

where $\Psi(x)$ is the Psi-function (also known as the digamma function). $\Psi(x)$ is evaluated using function `psi` from [9]. The second derivative is evaluated as:

$$\frac{d^2\Gamma(x)}{dx^2} = \Gamma(x)\Psi_1(x) \tag{11}$$

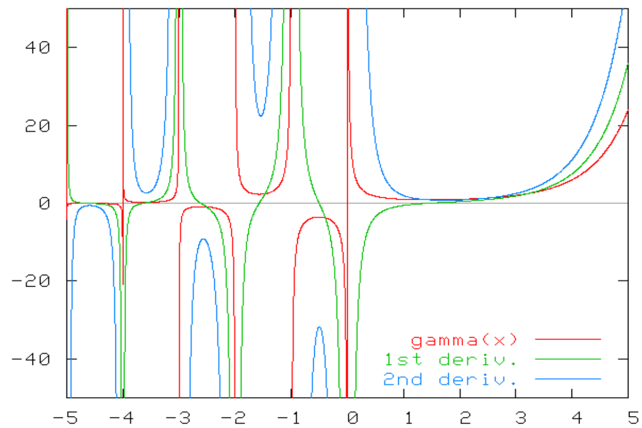

FIGURE 4. The `gamma(x)` function and its derivatives

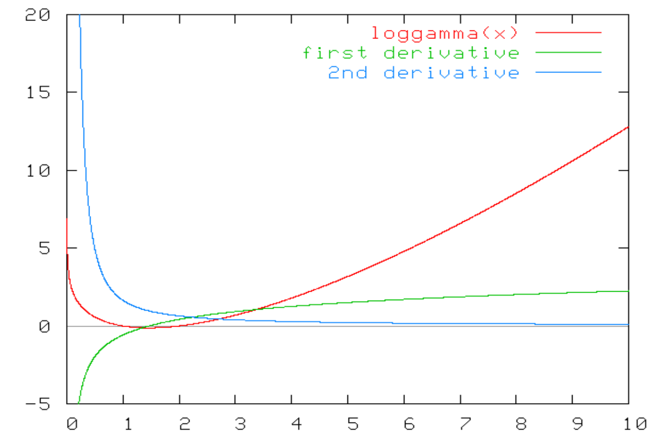

FIGURE 5. The `loggamma(x)` function and its derivatives

where $\Psi_1(x)$ is the trigamma function. $\Psi_1(x)$ is implemented using [36, 20]. For negative values $x < 0$ we use the identity[1]

$$\begin{aligned} \Gamma(x)\Gamma(-x) &= \frac{\pi}{x\sin(\pi x)} \Rightarrow \\ \ln\Gamma(x) + \ln\Gamma(-x) &= \ln\pi - \ln x - \ln\sin(\pi x) \Rightarrow \\ \Psi_1(x) + \Psi_1(-x) &= \frac{1}{x^2} + \pi^2\csc^2(\pi x) \end{aligned} \tag{12}$$

[1]Thanks to Herman Rubin, Department of Statistics, Purdue University for pointing this out to me

The derivatives of the `loggamma(.)` function are calculated directly as:

$$\frac{d \ln \Gamma(x)}{dx} = \Psi(x) \tag{13}$$
$$\frac{d^2 \ln \Gamma(x)}{dx^2} = \Psi_1(x)$$

The `gamma(x)` function is considered to be non-smooth and therefore has to be called using a DNLP solver instead of an NLP solver. The `loggamma(x)` function is smooth and can be called either by a DNLP or NLP solver.

4. The Incomplete Gamma function

The incomplete gamma function is a generalization of the gamma function [2, 42, 7]:

$$\gamma(x,a) = \frac{1}{\Gamma(a)} \int_0^x t^{a-1} e^{-t} dt \tag{14}$$

It is noted that the complement

$$\Gamma(x,a) = 1 - \gamma(x,a) = \frac{1}{\Gamma(a)} \int_x^\infty t^{a-1} e^{-t} dt \tag{15}$$

is also often referred to as the incomplete gamma function. Other definitions drop the constant $1/\Gamma(a)$. You will need to check carefully what definition is used.

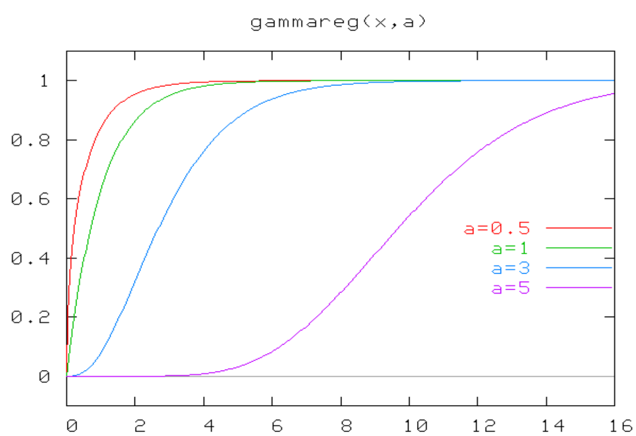

Figure 6. The incomplete gamma function

The implemented function for the incomplete gamma function `gammareg(x,a)` allows for a domain of $x > 0$ and $a > 0$[2]. Both derivatives with respect to x and to a are implemented. This means that both x and a can be variables when the function is used inside a model equation.

The implementation of the incomplete gamma function is based on [37]. The first and second derivatives are based on an algorithm from [31].

[2]To be precise: the function can be evaluated for $x = 0$ but the gradients require $x > 0$.

5. Distributions based on the gamma function

5.1. **The Gamma distribution.** The Gamma distribution[19, 25] has a distribution function:

$$F(x) = \gamma(x/\theta, k), x > 0 \tag{16}$$

with *shape parameter* k and *scale parameter* θ. We have $E(X) = k\theta$ and $Var(X) = k\theta^2$. The incomplete gamma function `gammareg(.)` can be used directly to evaluate the cdf (cumulative distribution function) of the Gamma distribution. As a result the plots in figure 6 can be interpreted directly as graphs of the gamma cdf.

Special cases of the Gamma distribution include the Exponential distribution (by choosing $k = 1$), the Erlang distribution (if k is an integer) and the Chi-square distribution (see below).

As an aside, the Excel function `GAMMADIST` is not always reliable. For instance a formula like `=+GAMMADIST(0.1,0.1,1,TRUE)` will return `#NUM!`. This is a known bug[12]. The accuracy of Excel's statistical functions has been discussed in several papers [27, 30, 11, 28].

Section 11.2 shows an example of maximum likelihood estimation of shape and location parameters of the Gamma distribution.

5.2. **The Chi-square distribution.** If $X_i \sim N(0, 1)$ and independent of each other, then

$$Y = \sum_{i=1}^{\nu} X_i^2 \sim \chi^2_\nu \tag{17}$$

has a Chi-square distribution with ν degrees of freedom.

The density function is:

$$f(x) = \frac{x^{\nu/2-1} e^{-x/2}}{\Gamma(\nu/2) 2^{\nu/2}} \tag{18}$$

The distribution function of the χ^2_ν distribution is:

$$F(x) = \gamma(x/2, \nu/2), x > 0 \tag{19}$$

The mean and variance are given by $E(X) = \nu$ and $Var(X) = 2\nu$.

6. The Beta function

The beta function is defined by[2, 44, 42]:

$$\begin{aligned} B(x, y) &= \int_0^1 t^{x-1}(1-t)^{y-1} dt \\ &= \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)} \\ &= B(y, x) \end{aligned} \tag{20}$$

The expression $\Gamma(x)\Gamma(y)/\Gamma(x + y)$ can be used directly in your model. GAMS will calculate the first and second derivatives using the chain rule for an expression like this. For convenience we have also implemented the beta function directly, which is called `beta(x,y)`. This function requires a `DNLP` model due to discontinuities when x or y are non-positive integers. For $x, y > 0$ the function is smooth.

In most cases you will use $x > 0$ and $y > 0$. For this case the function `logbeta(x,y)` may be more appropriate. This function implements

$$\ln B(x,y) = \ln\Gamma(x) + \ln\Gamma(y) - \ln\Gamma(x+y) \tag{21}$$

which does not use the quickly growing $\Gamma(.)$ function directly. The actual algorithm used to calculate `logbeta(x,y)` is taken from [16]. The derivatives are based on the identity 21.

7. The Incomplete Beta function

The Incomplete Beta function defined by[2, 42]:

$$I_x(a,b) = \frac{1}{B(a,b)}\int_0^x t^{a-1}(1-t)^{b-1}dt \tag{22}$$

is implemented as the `betareg(x,a,b)` function in GAMS.

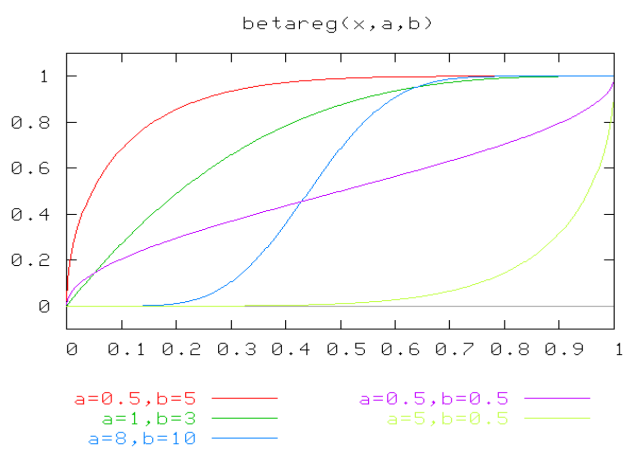

Figure 7. The incomplete beta function

We have:

$$\begin{aligned} I_0(a,b) &= 0 \\ I_1(a,b) &= 1 \end{aligned} \tag{23}$$

The function evaluator is based on [16, 4]. The derivatives $\partial I_x(a,b)/\partial a$, $\partial I_x(a,b)/\partial b$, $\partial^2 I_x(a,b)/\partial a^2$, $\partial^2 I_x(a,b)/\partial b^2$, $\partial^2 I_x(a,b)/\partial a\partial b$ are based on code from [3]. The other derivatives are evaluated as:

$$\begin{aligned} \frac{\partial I_x(a,b)}{\partial x} &= \frac{x^{a-1}(1-x)^{b-1}}{B(a,b)} \\ \frac{\partial^2 I_x(a,b)}{\partial x^2} &= \frac{(a-1)x^{a-2}(1-x)^{b-1} - (b-1)x^{a-1}(1-x)^{b-2}}{B(a,b)} \\ \frac{\partial^2 I_x(a,b)}{\partial x\partial a} &= x^{a-1}(1-x)^{b-1}\frac{\ln(x) - (\Psi(a) - \Psi(a+b))}{B(a,b)} \\ \frac{\partial^2 I_x(a,b)}{\partial x\partial b} &= x^{a-1}(1-x)^{b-1}\frac{\ln(1-x) - (\Psi(b) - \Psi(a+b))}{B(a,b)} \end{aligned} \tag{24}$$

These formulas have been verified with the Maxima CAS (Computer Algebra System) as follows:

```
[erwin@localhost erwin]$ maxima
GCL (GNU Common Lisp)  Version(2.5.0) Thu Dec  5 08:07:35 EST 2002
Licensed under GNU Library General Public License
Contains Enhancements by W. Schelter
Maxima 5.9.0 http://maxima.sourceforge.net
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
This is a development version of Maxima. The function bug_report()
provides bug reporting information.
(C1) incbeta(x,a,b):= (1/beta(a,b))*integrate(t**(a-1)*(1-t)**(b-1),t,0,x);

                              1                  a - 1        b - 1
(D1) incbeta(x, a, b) := ---------- INTEGRATE(t      (1 - t)     , t, 0, x)
                         BETA(a, b)
(C2) diff(diff(incbeta(x,a,b),x),x);

Is  x  positive, negative, or zero?

positive;
                          b - 1  a - 2                    b - 2  a - 1
           (a - 1) (1 - x)      x         (b - 1) (1 - x)      x
(D2)       --------------------------- - ---------------------------
                   BETA(a, b)                     BETA(a, b)
(C3) diff(diff(incbeta(x,a,b),x),a);

Is  x  positive, negative, or zero?

positive;
                                                                b - 1  a - 1
            b - 1  a - 1          (PSI (b + a) - PSI (a)) (1 - x)      x
     (1 - x)      x      LOG(x)        0            0
(D3) -------------------------- + -------------------------------------------
             BETA(a, b)                            BETA(a, b)
(C4) diff(diff(incbeta(x,a,b),x),b);

Is  x  positive, negative, or zero?

positive;
                        b - 1  a - 1
     LOG(1 - x) (1 - x)      x
(D4) ------------------------------
               BETA(a, b)

                                                                  b - 1  a - 1
                                    (PSI (b + a) - PSI (b)) (1 - x)      x
                                         0            0
                                  + -------------------------------------------
                                                     BETA(a, b)
(C5) quit();

[erwin@localhost erwin]$
```

The derivatives can only be calculated for $x > 0$ and $x < 1$.

8. Distributions based on the beta function

8.1. The beta distribution. The Beta distribution has the incomplete beta function as distribution function:

$$F(x) = I_x(p, q),\ 0 \leq x \leq 1 \tag{25}$$

where $p > 0$ and $q > 0$ are *shape parameters*. The mean and variance are given by

$$E(X) = \frac{p}{p+q}$$
$$Var(X) = \frac{pq}{(p+q)^2(p+q+1)} \tag{26}$$

The beta distribution is often used in measuring income distributions and poverty[15, 14].

8.2. **The generalized beta distribution.** The generalized beta distribution is defined over the interval $[a, b]$. The distribution function $F(x, a, b)$ can be written in terms of the distribution function of the *standard* beta distribution $F(x)$ as follows:

$$F(x, a, b) = F\left(\frac{x-a}{b-a}\right) \tag{27}$$

The cdf of the standard beta distribution $F(x)$ can be calculated directly using the incomplete beta function (see the previous paragraph). I.e.

$$F(x, a, b) = I_{(x-a)/(b-a)}(p, q),\ a \le x \le b \tag{28}$$

8.3. **The F distribution.** The F distribution is formed by the ratio of two chi-square distributions with degrees of freedom ν_1 and ν_2. The cdf is:

$$F(x) = I_y\left(\frac{\nu_1}{2}, \frac{\nu_2}{2}\right)$$
$$y = \frac{\nu_1 x}{\nu_2 + \nu_1 x} \tag{29}$$

or

$$F(x) = 1 - I_z\left(\frac{\nu_1}{2}, \frac{\nu_2}{2}\right)$$
$$z = \frac{\nu_2}{\nu_2 + \nu_1 x} \tag{30}$$

The mean and variance are given by:

$$E(X) = \frac{\nu_2}{\nu_2 - 1}$$
$$Var(X) = \frac{\nu_2(\nu_1 - 1)}{\nu_1(\nu_2 + 1)} \tag{31}$$

8.4. **Student's t distribution.** If $X_1, \ldots, X_n$ are independent normally distributed random variables $X_i \sim N(\mu, \sigma^2)$, then the quantity

$$T = \frac{\overline{X}_n - \mu}{S_n/\sqrt{n}} \tag{32}$$

has a t distribution with $\nu = n - 1$ degrees of freedom, where

$$\overline{X}_n = \frac{\sum_{i=1}^n X_i}{n} \tag{33}$$

is the sample mean and

$$S_n^2 = \frac{1}{n-1}\sum_{i=1}^n (X_i - \overline{X}_n)^2 \tag{34}$$

is the sample variance.

The Student's t distribution has a cdf given by:

$$\int_{-\infty}^{t} f(u)du = \begin{cases} 1 - \frac{1}{2} I_x(\nu/2, 1/2) & \text{if } t > 0 \\ \frac{1}{2} I_x(\nu/2, 1/2) & \text{otherwise} \end{cases} \tag{35}$$

where

$$x = \frac{\nu}{\nu + t^2} \tag{36}$$

For an example see section 11.8.

9. The Binomial function

The binomial function for integer arguments calculates *binomial coefficients* and is defined by

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad 0 \le k \le n \tag{37}$$

and gives the number of ways one can select k objects from a collection of n objects (the order in which objects are drawn is ignored). Often $\binom{n}{k}$ is pronounced as "n *choose* k". This function can be generalized for real arguments using:

$$\binom{x}{y} = \frac{\Gamma(x+1)}{\Gamma(y+1)\Gamma(x-y+1)} \tag{38}$$

where $\Gamma(.)$ is the *Gamma* function. This generalized binomial function is implemented in GAMS as the function `binomial(x,y)`.

10. The Binomial distribution

The binomial distribution describes the probability of k successes out of n Bernoulli trials, i.e.:

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{39}$$

The cdf of the binomial distribution is given by:

$$P(X \le k) = \sum_{j=0}^{k} \binom{n}{j} p^j (1-p)^{n-j} \tag{40}$$

which can be evaluated conveniently using the *incomplete Beta* function:

$$P(X \le k) = I_{1-p}(n-k, k+1). \tag{41}$$

For n large a binomial distribution can be approximated by a normal distribution $N(\mu, \sigma^2)$ with $\mu = np$ and $\sigma^2 = np(1-p)$. Using a normal approximation and a continuity correction we have:

$$P(X \le k) \approx \text{errorf}\left(\frac{k - np + 0.5}{\sqrt{np(1-p)}}\right). \tag{42}$$

See also section 11.5.

11. Some examples

11.1. Find minimum of gamma function. The minimum of the gamma function $\Gamma(x)$ for $x > 0$ is attained for $x^* = 1.46163...$[40]. This can be verified using a simple NLP. We also check whether minimizing `loggamma(x)` gives the same result.

```
$ontext

   Find minimum of y=gamma(x) and y=loggamma(x) for x>0

   Reference:

     Sloane, N. J. A., The On-Line Encyclopedia of Integer Sequences;
     Sequence A030169, http://www.research.att.com/projects/OEIS?Anum=A030169

$offtext


variables y1,y2,x1,x2;
equations y1def,y2def;

x1.lo = 0.1;
x1.l = 1;
x1.up = 5;

x2.lo = 0.1;
x2.l = 1;
x2.up = 5;

y1def.. y1 =e= gamma(x1);
y2def.. y2 =e= loggamma(x2);

model m1 /y1def/;
model m2 /y2def/;

solve m1 minimizing y1 using dnlp;
solve m2 minimizing y2 using nlp;

option decimals=8;
display x1.l,x2.l,y1.l,y2.l;

abort$(abs(x1.l-x2.l)>0.00001 or abs(log(y1.l)-y2.l)>0.00001) "inconsistent results";
```

The results are:

```
----     29 VARIABLE x1.L                  =   1.46163119
            VARIABLE x2.L                  =   1.46163174
            VARIABLE y1.L                  =   0.88560319
            VARIABLE y2.L                  =  -0.12148629
```

11.2. Maximum likelihood estimation of the Gamma distribution. Consider data collected on times between failures of air conditioning units in different aircraft[13]. We assume the times between failures are independent random variables with a Gamma distribution. Given a mean time between failures μ and a shape parameter β, the density function of the gamma distribution is[43]:

$$f(x) = \frac{(\beta/\mu)(\beta x/\mu)^{\beta-1}e^{-\beta x/\mu}}{\Gamma(\beta)} \tag{43}$$

The log likelihood function can now be written as:

$$L(\mu,\beta) = n\left[\ln\beta - \ln\mu - \ln\Gamma(\beta)\right] + \sum_{i=1}^{n}(\beta-1)\ln\left(\frac{\beta x_i}{\mu}\right) - \sum_{i=1}^{n}\frac{\beta x_i}{\mu} \tag{44}$$

We can maximize this function using the `loggamma(.)` function.

The method of moments estimator of β is

$$\hat{\beta} = \left(\frac{\hat{\mu}}{\hat{\sigma}}\right)^2 \tag{45}$$

which can be used as an (excellent) initial point for the optimization problem.

```
$ontext

   Maximum Likelihood estimation of parameters of the gamma distribution


   Erwin Kalvelagen, april 2004.


   Data from:
      COX, D. R. AND SNELL, E. J., (1981)
      Applied Statistics: Principles and Examples,
      London: Chapman and Hall.
   Example from:
      Luke Tierney, July 1989
      XLISP-STAT, A Statistical Environment Based on the XLISP Language (Version 2.0)
      Technical Report Number 528, University of Minnesota, School of Statistics




$offtext


set i 'observations' /i1*i29/

parameter x(i) 'times (in operating hours) between failures of airco units on several aircraft'
 /
  i1  90,  i2  10,  i3  60,  i4 186,  i5  61
  i6  49,  i7  14,  i8  24,  i9  56, i10  20
 i11  79, i12  84, i13  44, i14  59, i15  29
 i16 118, i17  25, i18 156, i19 310, i20  76
 i21  26, i22  44, i23  23, i24  62, i25 130
 i26 208, i27  70, i28 101, i29 208
/;

scalar n;
n = card(i);

scalar average;
average = sum(i, x(i))/n;

scalar stdev 'standard deviation';
stdev = sqrt(sum(i, sqr(x(i)-average))/(n-1));

display average,stdev;


variables beta,mu,like;
equations loglike;

loglike.. like =e= n*[log(beta)-log(mu)-loggamma(beta)] +
                   sum(i, (beta-1)*log(beta*x(i)/mu)) -
                   sum(i, beta*x(i)/mu);


*
* lowerbounds so log() and lngamma() are safe
*
beta.lo = 0.0001;
mu.lo = 0.0001;

*
* initial values using moments estimates
*
mu.l = average;
```

```
beta.l = sqr(average/stdev);


model m /loglike/;
solve m using nlp maximimizing like;
```

The resulting estimates for the parameters μ and β are:

```
                           LOWER          LEVEL          UPPER         MARGINAL

---- VAR beta               0.0001         1.6710        +INF       1.004352E-11
---- VAR mu                 0.0001        83.5172        +INF       1.944001E-13
---- VAR like              -INF         -155.3468        +INF             .
```

11.3. **Maximum likelihood estimation of the Beta distribution.** The log likelihood function of the beta distribution with parameters α and β is:
(46)

$$\ln L = n\left[\ln\Gamma(\alpha+\beta) - \ln\Gamma(\alpha) - \ln\Gamma(\beta)\right] + \sum_{i=1}^{n}(\alpha-1)\ln(x_i) + \sum_{i=1}^{n}(\beta-1)\ln(1-x_i)$$

This function can be implemented straightforwardly using the `loggamma` function.

The first two moments of the beta distribution, lead to two equations in two variables defining the method of moments estimator of α and β:

$$E(X) = \frac{\alpha}{\alpha+\beta}$$
$$Var(X) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{47}$$

or

$$\hat\alpha = \left[\frac{\hat\mu(1-\hat\mu)}{\hat\sigma^2} - 1\right]\hat\mu$$
$$\hat\beta = \left[\frac{\hat\mu(1-\hat\mu)}{\hat\sigma^2} - 1\right](1-\hat\mu) \tag{48}$$

```
$ontext

   Fitting of beta distribution through maximum likelihood

   Erwin Kalvelagen, april 2004

   Reference:
       Johnson, Kotz, and Balakrishnan, (1994),
       Continuous Univariate Distributions, Volumes I and II,
       2nd. Ed., John Wiley and Sons.

$offtext


set i 'cases' /i1*i75/;
parameter x(i) /

i1  4.973016e-01, i2  3.558841e-01, i3  2.419578e-02, i4  1.913753e-01, i5  4.919495e-01
i6  9.790016e-01, i7  3.856570e-01, i8  1.568263e-01, i9  8.040481e-01, i10 8.108720e-01
i11 6.016693e-01, i12 3.691279e-02, i13 9.454942e-01, i14 1.853702e-01, i15 3.496894e-01
i16 4.249933e-01, i17 9.900851e-01, i18 6.308701e-01, i19 4.474022e-02, i20 4.408432e-03
i21 3.718974e-03, i22 1.066217e-01, i23 5.304127e-01, i24 6.781648e-01, i25 6.206926e-02
i26 4.048511e-01, i27 4.941163e-01, i28 1.644695e-01, i29 2.285463e-02, i30 5.654344e-05
i31 2.657641e-01, i32 7.316988e-01, i33 6.789551e-01, i34 3.624824e-01, i35 7.429815e-03
i36 1.503384e-01, i37 7.314336e-01, i38 4.586442e-02, i39 4.060616e-02, i40 3.395101e-01
i41 9.269645e-01, i42 2.192909e-03, i43 2.511850e-02, i44 4.152490e-01, i45 1.612197e-01
i46 1.512879e-02, i47 1.381864e-01, i48 5.730967e-03, i49 1.185086e-01, i50 7.411310e-01
```

```
i51 1.564168e-02, i52 2.206906e-01, i53 9.836009e-01, i54 4.632388e-01, i55 9.968135e-01
i56 8.792355e-04, i57 9.692757e-01, i58 9.823214e-01, i59 1.248862e-01, i60 1.598848e-01
i61 9.561613e-02, i62 2.513807e-01, i63 4.435097e-01, i64 8.852468e-01, i65 1.149253e-02
i66 6.575999e-01, i67 8.236305e-01, i68 7.388426e-01, i69 6.382491e-01, i70 3.426699e-01
i71 1.244351e-01, i72 2.753017e-05, i73 1.625740e-01, i74 2.953334e-02, i75 8.739085e-02

/;

scalar n;
n = card(i);

scalar average;
average = sum(i, x(i))/n;

scalar stdev 'standard deviation';
stdev = sqrt(sum(i, sqr(x(i)-average))/(n-1));

variables alpha,beta,like;
equations loglike;

loglike.. like =e= n*[loggamma(alpha+beta)-loggamma(alpha)-loggamma(beta)] +
                   sum(i, (alpha-1)*log(x(i))) +
                   sum(i, (beta-1)*log(1-x(i)));


*
* lowerbounds so log() is safe
*
alpha.lo = 0.0001;
beta.lo = 0.0001;

*
* initial values using matching moments estimates
*
scalar tmp;
tmp = average*(1-average)/sqr(stdev) - 1;
alpha.l = tmp*average;
beta.l = tmp*(1-average);

display alpha.l,beta.l;

model m /loglike/;
solve m using nlp maximimizing like;

display alpha.l,beta.l;
```

11.4. **Plotting the incomplete gamma function.** This model was used to create figure 6 which reproduces figure 6.2.1 in [34] using calls to `gammareg(x,a)`. We use GNUPLOT [1] to produce the graph.

```
$ontext

   Create a plot of the incomplete gamma function, for parameters
   a = 0.5, 1, 3, and 10. This should reproduce figure 6.2.1 in
   the reference.

   Erwin Kalvelagen, april 2004

   References:
      William H. Press and Brian P. Flannery and Saul A. Teukolsky and
      William T. Vetterling, "Numerical Recipes in Fortran",
      Cambridge University Press, 2nd edition, 1992.

$offtext

set k /k1*k500/;
set a /a1*a4/;
parameter aval(a) /a1 0.5, a2 1, a3 3, a4 10/;
scalar xlo /0.0001/;
```

```
scalar xup /16/;
scalar n;     n = card(k);
scalar step;  step = (xup-xlo)/n;
parameter xpoint(k);    xpoint(k) = xlo + step*(ord(k)-1);
parameter ypoint(k,a);    ypoint(k,a) = gammareg(xpoint(k),aval(a));
display xpoint,ypoint;

file datafile /incgamma.dat/;
put datafile;
loop(k,
   put xpoint(k):17:9;
   loop(a,
       put ' ',ypoint(k,a):17:9
   );
   put /;
);
putclose;


file pltfile /incgamma.plt/;
put pltfile;
putclose
   'set yrange [-0.1:1.1]'/
   'set data style lines'/
   'set zeroaxis'/
   'set key 14,0.5'/
   'set title "gammareg(x,a)"'/
   'set term png medium color'/
   'set output "incgamma.png"'/
   'plot "incgamma.dat" using 1:2 title "a=0.5",'
        '"incgamma.dat" using 1:3 title "a=1",'
        '"incgamma.dat" using 1:4 title "a=3",'
        '"incgamma.dat" using 1:5 title "a=5"'

;

*
* linux, use Gnome image viewer to inspect the result
*
$if %system.platform% == WIN $goto windows
execute '=gnuplot incgamma.plt';
execute '=eog incgamma.png'
$exit

*
* windows, use registered viewer
*
$label windows
execute '=e:\applications\gnuplot\bin\wgnuplot.exe incgamma.plt';
execute '=ShellExecute incgamma.png'
```

11.5. **Evaluation of the binomial distribution.** This GAMS fragment will evaluate the cumulative distribution function of the binomial distribution in different ways:

- direct evaluation of the sum $P(X \le k) = \sum_{j=0}^{k} \binom{n}{j} p^j (1-p)^{n-j}$
- using the incomplete beta function $P(X \le k) = I_{1-p}(n-k, k+1)$
- using a normal approximation $N(\mu, \sigma^2)$ with $\mu = np$ and $\sigma^2 = np(1-p)$, with additionally a continuity correction

```
$ontext

   evaluation of binomial distribution function
   three alternatives:
     1. direct summation using binomial function
     2. equivalent: use regularized incomplete beta function
```

```
    3. normal approximation (with continuity correction)

   Erwin Kalvelagen, oct. 2006

$offtext


scalar n 'number of trials' /100/;
scalar p 'success probability' /0.3/;

set k 'number of successes' /k1*k5/;
parameter kval(k) /k1 20, k2 30, k3 40, k4 50, k5 60/;

set j /j0*j100/;
parameter jval(j) 'holds 0,1,2,..';
jval(j) = ord(j)-1;

parameter pr(k,*);
option pr:6:1:1;
pr(k,'k') = kval(k);
pr(k,'sum') =  sum(j$(jval(j)<=kval(k)),
       binomial(n,jval(j))*power[p,jval(j)]*power[1-p,n-jval(j)]);
pr(k,'beta') = betareg(1-p,n-kval(k),kval(k)+1);
pr(k,'norm.app.') = errorf([kval(k)-n*p+0.5]/sqrt[n*p*(1-p)]);
display n,p,pr;
```

The result is:

```
----     31 PARAMETER n                    =      100.000  number of trials
            PARAMETER p                    =        0.300  success probability

----     31 PARAMETER pr

             k         sum        beta   norm.app.

k1   20.000000    0.016463    0.016463    0.019083
k2   30.000000    0.549124    0.549124    0.543442
k3   40.000000    0.987502    0.987502    0.989027
k4   50.000000    0.999991    0.999991    0.999996
k5   60.000000    1.000000    1.000000    1.000000
```

11.6. **Economic selection of process mean.** A fairly complicated cost function is to be minimized here. In this problem[8, 33] from industrial engineering we seek a optimum process mean. The distribution of the product quality is often assumed to be normal, but there are arguments for choosing a beta distribution instead. In [33] a quadratic loss function is used, while in the model below we follow [8] with a linear loss function:

$$L(x) = \begin{cases} k_1(T-x) & x \le T \\ k_2(x-T) & x > T \end{cases} \tag{49}$$

where T is the ideal target value from the customer's point of view. The cost function will now be:

$$TC = \int_{a+\delta}^{T} k_1(T-x)f(x)dx + \int_{T}^{b+\delta} k_2(x-T)f(x)dx \tag{50}$$

where a and b are the minimum and maximum of the quality characteristic. The variable δ and the process mean μ are related as:

$$\mu = \delta + a + (b-a)\frac{\alpha}{\alpha+\beta} \tag{51}$$

Some tedious transformations lead to

$$
\begin{aligned}
TC =& k_1 T I_y(\alpha,\beta) \\
& - k_1 \left\{(\delta+a) I_y(\alpha,\beta) + (b-a) I_y(\alpha+1,\beta)\Upsilon\right\} \\
& + k2 \left\{(\delta+a)\left[1 - I_y(\alpha,\beta)\right] + (b-a)\left[1 - I_y(\alpha+1,\beta)\Upsilon\right]\right\} \\
& - k_2 T \left[1 - I_y(\alpha,\beta)\right]
\end{aligned} \tag{52}
$$

where

$$
\begin{aligned}
y &= \frac{T-\delta-a}{b-a} \\
\Upsilon &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \cdot \frac{\Gamma(\alpha+1)\Gamma(\beta)}{\Gamma(\alpha+\beta+1)}
\end{aligned} \tag{53}
$$

It is noted that we can use a parameter called `beta`. This parameter will hide the function `beta(.)`.

```
$ontext

    Find the optimal process mean when the quality characteristic
    follows a Beta distribution and using a linear quality loss.

    Erwin Kalvelagen, april 2004

    References:
         Chung-Ho Chen, Chao-Yu Chou,
         "Determining the Optimum Process Mean under a Beta Distribution",
         Journal of the Chinese Institute of Industrial Engineers,
         Vol. 18, No.3, pp. 27-32, 2003

         M.D. Phillips and B.-R. Cho,
         "A Nonlinear model for determining the most economic process mean
         under a beta distribution", International Journal of Reliability,
         quality and Safety Engineering, vol.7, pp. 61-74, 2000

$offtext

scalars
   a 'minimum value of quality characteristic'   /113/
   b 'maximum value of quality characteristic'   /119/
   alpha 'shape parameter' /2/
   beta  'shape parameter' /4/
   T 'target value' /115/
   k1 'quality loss coefficient when x<T' /2/
   k2 'quality loss coefficient when x>T' /3/
;


scalars g1,g2,g3;

g1 = gamma(alpha+beta)/(gamma(alpha)*gamma(beta));
g2 = gamma(alpha+1)*gamma(beta)/gamma(alpha+beta+1);
g3 = g1*g2;


variables
   TC 'Total expected cost per unit'
   delta 'location parameter'
   y  'transformation'
;

equations
   tcdef 'cost model'
   ydef
;

tcdef..  tc =e= k1*T*betareg(y,alpha,beta)
                - k1*{(delta+a)*betareg(y,alpha,beta)
```

```
                    +(b-a)*betareg(y,alpha+1,beta)*g3}
              + k2*{(delta+a)*[1-betareg(y,alpha,beta)]
                    +(b-a)*[1-betareg(y,alpha+1,beta)*g3]}
              - k2*T*[1-betareg(y,alpha,beta)];

ydef..   y =e= (T-delta-a)/(b-a);

y.lo = 0.0001;
y.up = 0.9999;

y.l = 0.5;

model m /all/;
solve m using nlp minimizing tc;
```

The optimal solution is:

```
                           LOWER          LEVEL          UPPER         MARGINAL

---- VAR TC                -INF           14.1661        +INF             .
---- VAR delta             -INF           -0.1899        +INF             .
---- VAR y                  0.0001         0.3650         0.9999    -3.33415E-10
```

which is actually slightly better than reported in [8].

11.7. **Quantum mechanics.** In this application of quantum mechanics we try to find the ground state or minimum energy state by minimizing the expectation of the hamiltonian[32]:

$$
\begin{aligned}
I^{(0)}(\alpha,n) &\equiv \langle \psi^{(0)}(n,\alpha)|H|\psi^{(0)}(n,\alpha)\rangle \\
&= \frac{n^2}{2}\frac{\Gamma(2-\frac{1}{2n})}{\Gamma(\frac{1}{2n})}\alpha^{1/n} + \frac{1}{2}\frac{\Gamma(\frac{3}{2n})}{\Gamma(\frac{1}{2n})}\alpha^{-1/n} + g\frac{\Gamma(\frac{5}{2n})}{\Gamma(\frac{1}{2n})}\alpha^{-2/n}
\end{aligned}
\tag{54}
$$

where g is a constant. When $n = 1$ we have the standard Gaussian trial function. The model below calculates both the optimal values of α and n for the Gaussian and post-Gaussian case.

```
$ontext

   An application from quantum mechanics:
   Find energy eigenvalues of the anharmonic oscillator with g=1
   in the Gaussian and Post-Gaussian variational methods.

   Erwin Kalvelagen, May 2004

    Reference:
        Akihiro Ogura, "Post-Gaussian variational method for quantum
        anharmonic oscillator", Laboratory of Physics, College of Science
        and Technology, Nihon University, 1999.
        arXiv:physics/9905056 v1 28 May 1999

$offtext

variables
   ham 'expected value of hamiltonian'
   alpha 'variational parameter'
   n 'variational parameter (n=1: Gaussian trial function)'
;

equation
   hamiltonian
;

scalar g /1/;

hamiltonian..

   ham =e= (sqr(n)/2)*(gamma(2-1/(2*n))/gamma(1/(2*n)))*(alpha**(1/n))
```

```
          +(1/2)*(gamma(3/(2*n))/gamma(1/(2*n)))*(alpha**(-1/n))
          +g*(gamma(5/(2*n))/gamma(1/(2*n)))*(alpha**(-2/n));


alpha.lo = 0.0001; alpha.up = 10; alpha.l=1;

*
* gaussian variational method
*

n.fx = 1;

model m /hamiltonian/;
solve m minimizing ham using dnlp;

parameter results(*,*);
results('Gaussian','Ground') = ham.l;
results('Gaussian','alpha') = alpha.l;
results('Gaussian','n') = n.l;


*
* post-gaussian variational method
*
n.lo = 0.001; n.up = 10;
solve m minimizing ham using dnlp;

results('Post-Gaussian','Ground') = ham.l;
results('Post-Gaussian','alpha') = alpha.l;
results('Post-Gaussian','n') = n.l;

option decimals = 6;
display results;
```

The result is:

```
----     62 PARAMETER results

                  Ground       alpha           n

Gaussian        0.812500    2.000000    1.000000
Post-Gaussian   0.804903    1.866470    1.134934
```

11.8. **Linear Regression.** Linear regression estimators for the statistical model

$$y = X\beta + \epsilon \tag{55}$$

are given by:

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{56}$$

Statistical packages often provide additional information on these estimates, such as standard errors and t-values. Such statistics can be calculated from:

$$\begin{aligned} \hat{\sigma} &= \sqrt{\frac{\text{SSR}}{n-p}} \\ &= \sqrt{\frac{\sum_{i=1}^{n} \hat{\epsilon}_i^2}{n-p}} \end{aligned} \tag{57}$$

where n is the number of observations and p is the number of parameters to estimate (i.e. the number of elements in β). Furthermore the standard errors are given by:

$$\begin{aligned} \text{Var} &= \hat{\sigma}^2 \, \text{diag}(X^T X)^{-1} \\ \text{SE}_i &= \sqrt{\text{Var}_i} \end{aligned} \tag{58}$$

and the t values by:

$$t_i = \frac{\hat{\beta}_i}{\text{SE}_i} \tag{59}$$

food	income	food	income
9.46	25.83	17.77	71.98
10.56	34.31	22.44	72.00
14.81	42.50	22.87	72.23
21.71	46.75	26.52	72.23
22.79	48.29	21.00	73.44
18.19	48.77	37.52	74.25
22.00	49.65	21.69	74.77
18.12	51.94	27.40	76.33
23.13	54.33	30.69	81.02
19.00	54.87	19.56	81.85
19.46	56.46	30.58	82.56
17.83	58.83	41.12	83.33
32.81	59.13	15.38	83.40
22.13	60.73	17.87	91.81
23.46	61.12	25.54	91.81
16.81	63.10	39.00	92.96
21.35	65.96	20.44	95.17
14.87	66.40	30.10	101.40
33.00	70.42	20.90	114.13
25.19	70.48	48.71	115.46

TABLE 3. A household food expenditure data set

The following example is using a dataset from [23] which is reproduced in table 3. When we run a linear regression through the statistical system R [35] we get:

```
[erwin@fedora specfun]$ R

R : Copyright 2004, The R Foundation for Statistical Computing
Version 1.9.1  (2004-06-21), ISBN 3-900051-00-3

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for a HTML browser interface to help.
Type 'q()' to quit R.

> expenditure <- c(9.46, 10.56, 14.81, 21.71, 22.79, 18.19, 22.00, 18.12, 23.13,
+           19.00, 19.46, 17.83, 32.81, 22.13, 23.46, 16.81, 21.35, 14.87, 33.00,
+           25.19, 17.77, 22.44, 22.87, 26.52, 21.00, 37.52, 21.69, 27.40, 30.69,
+           19.56, 30.58, 41.12, 15.38, 17.87, 25.54, 39.00, 20.44, 30.10, 20.90,
+           48.71)
> income      <- c(25.83, 34.31, 42.50, 46.75, 48.29, 48.77, 49.65, 51.94, 54.33,
+           54.87, 56.46, 58.83, 59.13, 60.73, 61.12, 63.10, 65.96, 66.40, 70.42,
+           70.48, 71.98, 72.00, 72.23, 72.23, 73.44, 74.25, 74.77, 76.33, 81.02,
+           81.85, 82.56, 83.33, 83.40, 91.81, 91.81, 92.96, 95.17, 101.40, 114.13,
```

```
+          115.46)
> fm <- lm(expenditure ~ income)
> summary(fm)

Call:
lm(formula = expenditure ~ income)

Residuals:
    Min      1Q  Median      3Q     Max
-12.990  -3.561  -1.082   3.214  14.511

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.38322    4.00836   1.842 0.073296 .
income       0.23225    0.05529   4.200 0.000155 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.845 on 38 degrees of freedom
Multiple R-Squared: 0.3171,     Adjusted R-squared: 0.2991
F-statistic: 17.64 on 1 and 38 DF,  p-value: 0.0001551

> quit()
Save workspace image? [y/n/c]: n
[erwin@fedora specfun]$
```

The column `Pr(>|t|)` gives the probability of getting a t value larger than the one obtained if the value of the regression coefficient would be zero. This is also known as the p-value. If this value is below 0.05 then the null hypothesis "the coefficient is zero" is rejected at the 5% level. The two-tailed test is defined by

$$p = 2P(t_i > |\text{tstat}|) \tag{60}$$

where |tstat| is the absolute value of the calculated test statistic. The cdf of the t distribution is used to calculate this quantity. This distribution is also called Student's t distribution, after the pseudonym used by William Sealey Gosset[22]. The cdf is given by:

$$P(X \le t) = 1 - \frac{1}{2} I_x(\nu/2, 1/2) \text{ for } 0 \le t < \infty \tag{61}$$

where ν is the number of degrees of freedom and

$$x = \frac{\nu}{\nu + t^2} \tag{62}$$

and $I_x(a,b)$ is the incomplete beta function.

The following GAMS model will print similar results as the R run:

```
----    145 PARAMETER results

            Estimate  Std. Error     t value    Pr(>|t|)

constant       7.383       4.008       1.842       0.073
income         0.232       0.055       4.200 1.551364E-4
```

It is noted that when we form $(X^T X)$ and calculate $(X^T X)^{-1}$ using a small LP numerical problems can occur. More stable methods will not form $(X^T X)$ explicitly.

```
$ontext

   Linear Regression Statistics

   Erwin Kalvelagen, 2004

$offtext

set i 'observations' /i1*i40/;
```

```
set j 'explanatory variables' /constant,income/;

* cross-section data: weekly household expenditure on food and
* weekly household income from Griffiths, Hill and Judge,
* 1993, Table 5.2, p. 182.

table data(i, *)
        expenditure income
i1        9.46       25.83
i2       10.56       34.31
i3       14.81       42.50
i4       21.71       46.75
i5       22.79       48.29
i6       18.19       48.77
i7       22.00       49.65
i8       18.12       51.94
i9       23.13       54.33
i10      19.00       54.87
i11      19.46       56.46
i12      17.83       58.83
i13      32.81       59.13
i14      22.13       60.73
i15      23.46       61.12
i16      16.81       63.10
i17      21.35       65.96
i18      14.87       66.40
i19      33.00       70.42
i20      25.19       70.48
i21      17.77       71.98
i22      22.44       72.00
i23      22.87       72.23
i24      26.52       72.23
i25      21.00       73.44
i26      37.52       74.25
i27      21.69       74.77
i28      27.40       76.33
i29      30.69       81.02
i30      19.56       81.85
i31      30.58       82.56
i32      41.12       83.33
i33      15.38       83.40
i34      17.87       91.81
i35      25.54       91.81
i36      39.00       92.96
i37      20.44       95.17
i38      30.10      101.40
i39      20.90      114.13
i40      48.71      115.46
;

alias (j,jj,k);

*
* form X and y
*
parameter x(i,j), y(i);
x(i,'constant') = 1;
x(i,'income') = data(i,'income');
y(i) = data(i,'expenditure');

*
* form X'X and X'y
*
parameter xx(j,jj), xy(j);
xx(j,jj) = sum(i, x(i,j)*x(i,jj));
xy(j) = sum(i, x(i,j)*y(i));


*
* calculate inv(X'X)
*
variable
```

```
   dummy         'dummy objective variable'
   invxx(j,jj)   'inverse of xx'
;
equation
   edummy        'dummy objective function'
   invert(j,jj)  'calculate inverse matrix'
;
parameter identity(j,jj);
identity(j,j) = 1;

edummy..        dummy =e= 0;
invert(j,jj)..  sum(k, xx(j,k)*invxx(k,jj)) =e= identity(j,jj);

model inv /edummy,invert/;
solve inv using lp minimizing dummy;


*
* calculate estimates b = inv(X'X) X'y
*
parameter b(j);
b(j) = sum(k, invxx.l(j,k)*xy(k));

*
* calculate residuals
*
parameter res(i);
res(i) = sum(j, x(i,j)*b(j)) - y(i);
scalar rss 'residual sum of squares';
rss = sum(i, sqr(res(i)));

*
* calculate standard errors
*
scalar df;
df = card(i)-card(j);
scalar sigma;
sigma = sqrt(rss/df);
parameter se(j);
se(j) = sigma*sqrt(invxx.l(j,j));

*
* calculate t values
*
parameter tval(j);
tval(j) = b(j)/se(j);

*
* calculate p values
*
parameter pt(j);
pt(j) = 0.5*betareg(df/(df+sqr(tval(j))),df/2,0.5);
parameter pval(j);
pval(j) = 2*pt(j);

*
* prepare results table
*
parameter results(j,*);
results(j,'Estimate') = b(j);
results(j,'Std. Error') = se(j);
results(j,'t value') = tval(j);
results(j,'Pr(>|t|)') = pval(j);
display results;
```

11.9. **Stirling's formula.** The gamma function can be approximated by

$$\Gamma(z) \sim e^{-z} z^{z-1/2} \sqrt{2\pi} \left(1 + \frac{1}{12z} + \frac{1}{288z^2} - \frac{139}{51840z^3} - \frac{571}{2488320z^4} + \cdots\right) \tag{63}$$

This series is known as Stirling's formula[2]. For a large number of coefficients see [41]. This can be used inside a GAMS model, e.g.:

```
calc_beta.. beta =e=
    [exp(-a)*(a**(a-0.5))*sqrt(2*pi)*(1+1/(12*a)+1/(288*sqr(a))-139/(51840*power(a,3))
        -571/(2488320*power(a,4))+163879/(209018880*power(a,5)))]
   *[exp(-b)*(b**(b-0.5))*sqrt(2*pi)*(1+1/(12*b)+1/(288*sqr(b))-139/(51840*power(b,3))
        -571/(2488320*power(b,4))+163879/(209018880*power(b,5)))]
   /[exp(-a-b)*((a+b)**(a+b-0.5))*sqrt(2*pi)*(1+1/(12*(a+b))+1/(288*sqr(a+b))
        -139/(51840*power(a+b,3))-571/(2488320*power(a+b,4))
        +163879/(209018880*power(a+b,5)))];
```

11.10. **Generating random chi-square numbers.** In some cases the nonlinear solver capabilities of GAMS can be used to write quick-and-dirty random number generators. Consider the chi-square distribution. The cumulative distribution function of the chi-square distribution with ν degrees of freedom is given by

$$F(x) = \gamma(\frac{x}{2}, \frac{\nu}{2}) \tag{64}$$

where $\gamma(.)$ is the incomplete gamma function. The algorithm

1. Generate a uniform variate $U \sim U(0,1)$
2. Solve $F(x) = U$ for x

is readily implemented in GAMS:

```
$ontext

     Random number generation from Chi-Square distribution.

     Erwin Kalvelagen, march 2005

     Algorithm:
       1. generate u from U(0,1)
       2. solve F(x) = u where F(x) is the
          cdf of the chi-square distribution.

     Note: cdf of chi-square distribution with v degrees of freedom is:

          F(x) = gammareg(x/2,v/2)

$offtext

scalar nu 'degrees of freedom' /5/;

set i /i1*i10/;
parameter chisquare(i) 'chi square variates';

parameter u(i);
u(i) = uniform(0,1);

variable x(i);
equation f(i);

f(i).. gammareg(x(i)/2,nu/2) =e= u(i);

model m /f/;
x.lo(i) = 0;
x.l(i) = 1;
x.up(i) = 100;
solve m using cns;

chisquare(i) = x.l(i);

display u,chisquare;
```

11.11. **Generating random numbers from the beta distribution.** The approach of the previous section can be applied to the beta distribution, with density function

$$f(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)} \tag{65}$$

where $B(\alpha,\beta)$ is the beta function:

$$B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \tag{66}$$

The distribution function is equal to the (regularized) incomplete beta function:

$$F(x) = I_x(\alpha,\beta) \tag{67}$$

The resulting generator can look like:

```
$ontext

     Random number generation from the beta distribution.

     Erwin Kalvelagen, april 2005

     Algorithm:
       1. generate u from U(0,1)
       2. solve F(x) = u where F(x) is the
          cdf of the beta distribution.

     Note: the cdf of the beta distribution with parameters a,b is:

          F(x) = betareg(x,a,b)

$offtext

scalar a /1/;
scalar b /3/;

set i /i1*i10/;
parameter beta(i) 'beta variates';

parameter u(i);
u(i) = uniform(0,1);

variable x(i);
equation f(i);

f(i).. betareg(x(i),a,b) =e= u(i);

model m /f/;
x.lo(i) = 1.0e-6;
x.l(i) = 0.5;
x.up(i) = 1-1.0e-6;
solve m using cns;

beta(i) = x.l(i);

display u,beta;
```

11.12. **DEA bootstrapping.** The model below implements a bootstrapping algorithm for the Data Envelopment Analysis Problem. DEA is a methodology to estimate efficient frontiers [5, 6, 18].

Bootstrapping[17, 39] is used to provide additional information for statistical inference. The following model from [48] implements a resampling strategy from [38]. Two thousand bootstrap samples are formed, each resulting in a DEA model of 100 small LP's. In this example we batch the DEA models together in a single large

LP, so that we only have to solve 2,000 models instead of 200,000. The formulation trick is explained in [26].

```
$ontext

   DEA bootstrapping example

   Erwin Kalvelagen, october 2004

   References:

     Mei Xue, Patrick T. Harker
     "Overcoming the Inherent Dependency of DEA Efficiency Scores:
     A Bootstrap Approach", Tech. Report, Department of Operations and
     Information Management, The Wharton School, University of Pennsylvania,
     April 1999

     http://opim.wharton.upenn.edu/~harker/DEAboot.pdf

$offtext


sets
   i 'hospital (DMU)' /h1*h100/
   j 'inputs and outputs' /
     FTE     'The number of full time employees in the hospital in FY 1994-95'
     Costs   'The expenses of the hospital ($million) in FY 1994-95'
     PTDAYS  'The number of the patient days produced by the hospital in FY 1994-95'
     DISCH   'The number of patient discharges produced by the hospital in FY 1994-95'
     BEDS    'The number of patient beds in the hospital in FY 1994-95'
     FORPROF 'Dummy variable, one if it is for-profit hospital, zero otherwise'
     TEACH   'Dummy variable, one if it is teaching hospital, zero otherwise'
     RES     'The number of the residents in the hospital in FY 1994-95'
     CONST   'Constant term in regression model'
   /
   inp(j) 'inputs' /FTE,Costs/
   outp(j) 'outputs' /PTDAYS,DISCH/
;

table data(i,j)

          FTE      Costs        PTDAYS   DISCH   BEDS FORPROF TEACH   RES

  h1     1571.86    174          71986   12665   365
  h2      816.54     69.9        53081    5861   224
  h3      533.74     61.7        25030    4951   286     1
  h4      805.2      75.4        34163   11877   256
  h5     3908.1     396         187462   42735   829            1    136.8
  h6      727.72     63.9        31330    8402   194
  h7     2571.75    220         130077   26877   620            1     42.81
  h8      521        89.1        43390    8598   290     1
  h9      718        50          27896    6113   150            1     23.21
  h10    1504.85    121          75941   16427   393
  h11    1234.49     84.6        57080   14180   317
  h12     873        68.8        48932   12060   281
  h13    1067.17     85.8        50436   11317   278
  h14     668        47.5        67909    6235   244
  h15     452.35     36.4        25200    6860   155     1      1     13.31
  h16    1523        97.4        59809   13180   394
  h17    3152       198         108631   22071   578            1    195.67
  h18     871.96     30.7        17925    4605   160
  h19    2901.86    290         130004   24133   549            1    126.89
  h20     902.4      78.2        35743    8664   236            1     12.08
  h21     194.69     10.9        15555    1530   132
  h22     713.51     62.6        32558    8966   138
  h23     557.36     23.8        12728    2291   276     1
  h24    2259.2     120          74061   12942   348            1     14.52
  h25     462.22     32.4        28886    6101   134
  h26    1212.1      97.3        74194   12681   342
  h27    2391.94    192          89843   18396   336            1    229.19
  h28    1637       162          80468   21345   415
  h29     501        37.9        26813    4594   166     1
```

```
  h30    412.1       40.2        23217    6044   160     1
  h31    738.56      27          11514    3052   144     1
  h32    414.1       35.7        55611    4354   200
  h33   1097        105          59443   13101   242            1      26.32
  h34    742         62.8        42542    8739   172
  h35   1010         97.1        47246   12073   269            1       1.1
  h36    440.6       34.2        30773    4305   201
  h37   1203.3       85.4        50710   11470   247            1      13.82
  h38   2558.01     195         128450   20441   571            1       5.42
  h39    215.45       8.409936   65743     578   238
  h40    599.3       30.4        23299    5338   173
  h41    480.55      29.5        34279    6560   169     1
  h42    634.51      29.9        27157    5198   141
  h43   1211.9       91.4        90008   17666   320            1       6.25
  h44    285.5       23.9        16473    2873   135
  h45   1030.36      67.1        43486    9467   235            1       6.44
  h46   1374.81      95.5        74279   11862   284
  h47    953.56      49.8        47934   10553   207
  h48    561.11      41.7        24800    5498   132
  h49    644         57.1        39663    8604   260
  h50    376.55      19.6        22003    4759   143
  h51    404.79      32.8        27566    7871   190     1
  h52    397.9       29.4        26072    4248   170
  h53    374.2        3.944649    4179     819   156
  h54   1702        100         114603   17235   438            1      11.81
  h55    148.09       5.013379   51660     771   172
  h56    253.48      16.9        17599    4044   178
  h57   1445.68      99.3        81041   12912   475            1      17.53
  h58    414.1       26.5        20432    4068   129
  h59    642.58      48.5        42733    5983   181     1
  h60    203.75      13          16923    3467   146     1
  h61    421.8       18.3        16179    2840   160
  h62    320.62      17.3        18882    3370   160
  h63    679.79      25.6        27561    4447   308            1      11.33
  h64   2382        226         166559   26019   787            1       7.08
  h65    559.29      58.1        40534    8806   342     1
  h66    568.15      35          37120    7242   158
  h67   2408.04     155          70392    9538   266            1     111.33
  h68    632.34      54.6        37228    6359   175
  h69    917.22      55.2        42135    7294   215
  h70    554.34      56.9        32352    3320   205            1       1
  h71    780         75.9        39213    7154   172
  h72    663.82      56.9        34180    5284   200
  h73   1424        146         107457   18198   432            1       2.75
  h74    313         20.7        20110    5967   165     1
  h75    778         78.4        51496   12302   390
  h76    863.37      62          50957   10557   228
  h77   3509.12     290         109673   19213   469            1     290.53
  h78   1593.82     152          82400   17707   474            1      11.64
  h79    466         40.1        30647    7265   164     1
  h80    666.38      48.2        28048    5182   153
  h81    998.8      121          45513    6855   238            1      88.86
  h82   1018         98.2        61176   11386   350
  h83   3238.28     326         122118   19068   514            1     146.33
  h84   1431.1      107          48900   10623   208
  h85   1735.99     273          84118   16458   278            1     158.4
  h86   1769        190         105741   19256   478            1       0.93
  h87    484.56      36.2        24070    6464   125
  h88    204.7       13.9        28137    1615   135     1
  h89   1706.58     287          75153   13465   367            1      91.56
  h90   1029.11      71.9        49993    6690   252            1       4
  h91   1167.2      111          75004   21334   350
  h92   1657.58     116          77753   17528   413
  h93   1017.16      88.5        64147   11135   316
  h94   1532.7      153          99998   17391   395            1       4.8
  h95   1462        113         119107   16053   484            1       0.5
  h96   1133.8      109          55540   15566   355            1       8.51
  h97    609         48.2        71817    5639   376            1       1
  h98    301.31      20.2        43214    2153   141
  h99   1930.08     201          87197   19315   418
  h100  1573.3      177          88124   19661   458            1      69.71
;
```

```
data(i,'CONST') = 1;


*------------------------------------------------------------------------------
* PHASE 1: Estimation of b(j)
*
* Run standard Constant Returns to Scale (CCR) Input-oriented DEA model
* followed by linear regression OLS estimation
*------------------------------------------------------------------------------


*
* this is the standard DEA model
* instead of 100 small models we solve one big model, see
* http://www.gams.com/~erwin/dea/dea.pdf
*
parameter
  x(inp,i)  'inputs of DMU i'
  y(outp,i) 'outputs of DMU i'
;

alias(i,j0);
positive variables
  v(inp,j0)    'input weights'
  u(outp,j0)   'output weights'
;
variable
  eff(j0) 'efficiency'
  z 'objective variable'
;

equations
  objective(j0)  'objective function: maximize efficiency'
  normalize(j0)  'normalize input weights'
  limit(i,j0)    "limit other DMU's efficiency"
  totalobj
;

totalobj..      z =e= sum(j0, eff(j0));
objective(j0).. eff(j0) =e= sum(outp, u(outp,j0)*y(outp,j0));
normalize(j0).. sum(inp, v(inp,j0)*x(inp,j0)) =e= 1;
limit(i,j0)..   sum(outp, u(outp,j0)*y(outp,i)) =l= sum(inp, v(inp,j0)*x(inp,i));

model dea /totalobj,objective, normalize, limit/;

alias (i,iter);

x(inp,i) = data(i,inp);
y(outp,i) = data(i,outp);

option limrow=0;
option limcol=0;
dea.solprint=2;
dea.solvelink=2;

solve dea using lp maximizing z;
abort$(dea.modelstat<>1) "LP was not optimal";

display
  "----------------------------------- DEA MODEL ------------------------",
  eff.l;

*
* now solve the regression problem
*   efficiency = b0 + b1*BEDS + b2*FORPROF + b3*TEACH + b4*RES
* Use b = inv(X^TX) X^Ty
* Standard errors are sigma^2 inv(X^TX)
* See http://www.gams.com/~erwin/stats/ols.pdf
*

set e(j) 'explanatory variables' /BEDS,FORPROF,TEACH,RES,CONST/;
```

```
*
* calculate inv(X^TX)
*
alias(e,ee,eee);
parameter XX(e,ee) 'matrix (X^TX)';
XX(e,ee) = sum(i,data(i,e)*data(i,ee));
parameter Xy(e) 'X^Ty';
Xy(e) = sum(i, data(i,e)*eff.l(i));
parameter ident(e,ee) 'Identity matrix';
ident(e,e)=1;

variable
   invXX(e,ee) 'matrix inv(X^TX)'
   dummy
;

equation
   invert(e,ee)
   edummy
;

invert(e,ee).. sum(eee, XX(e,eee)*invXX(eee,ee)) =e= ident(e,ee);
edummy.. dummy=e=0;
model matinv /invert,edummy/;
matinv.solprint=2;
matinv.solvelink=2;
solve matinv using lp minimizing dummy;

*
* calculate estimates and standard errors
*

parameter b(e);
b(e) = sum(ee, invXX.l(e,ee)*Xy(ee));

parameter resid(i) 'residuals';
resid(i) =  eff.l(i) - sum(e,b(e)*data(i,e));
scalar rss 'residual sum of squares';
rss = sum(i, sqr(resid(i)));


*
* calculate standard errors
*

scalar df 'degrees of freedom';
df = card(i)-card(e);
scalar sigma_squared 'variance of estimate';
sigma_squared = rss/df;
parameter variance(e,ee);
variance(e,ee) = sigma_squared*invXX.l(e,ee);
parameter se(e) 'standard error';
se(e) = sqrt(variance(e,e));

parameter tval(e) "t statistic";
tval(e) = b(e)/se(e);

parameter pval(e) "p-values";

*
*  pvalue = 2 * pt( abs(tvalue), df)
*         = 2 * 0.5 * pbeta( df / (df + sqr(abs(tvalue))), df/2, 0.5)
*         = betareg( df / (df+sqr(tvalue)), df/2, 0.5)
*
pval(e) = betareg( df / (df+sqr(tval(e))), df/2, 0.5);

parameter ols(e,*);
ols(e,'estimates') = b(e);
ols(e,'std.error') = se(e);
ols(e,'t value') = tval(e);
ols(e,'p value') = pval(e);
```

```
display
  "----------------------------------- OLS MODEL -----------------------",
  ols;


*---------------------------------------------------------------------------
* PHASE 2: BOOTSTRAP algorithm
*---------------------------------------------------------------------------

set s 'sample' /sample1*sample2000/;


parameter bs(s,i) 'bootstrap sample';
bs(s,i) = trunc( uniform(1,card(i)+0.999999999) );
*display bs;
* sanity check:
loop((s,i),
   abort$(bs(s,i)<1) "Check bs for entries < 1";
   abort$(bs(s,i)>card(i)) "Check bs for entries > card(i)";
);

alias(i,ii);
set mapbs(s,i,ii);
mapbs(s,i,ii)$(bs(s,i) = ord(ii)) = yes;
* this mapping says the i'th sample data record is the ii'th record
* in the original data (for sample s)

loop((s,i),
  abort$(sum(mapbs(s,i,ii),1)<>1) "mapbs is not unique";
);

parameter data_sample(i,j);

parameter sb(s,e) 'b(e) for each sample s';

loop(s,

*
* solve dea model
*

    data_sample(i,j) = sum(mapbs(s,i,ii),data(ii,j));
    x(inp,i) = data_sample(i,inp);
    y(outp,i) = data_sample(i,outp);

    solve dea using lp maximizing z;
    abort$(dea.modelstat<>1) "LP was not optimal";

*
* solve OLS model
*
    XX(e,ee) = sum(i,data_sample(i,e)*data_sample(i,ee));
    Xy(e) = sum(i, data_sample(i,e)*eff.l(i));
    solve matinv using lp minimizing dummy;
    sb(s,e) = sum(ee, invXX.l(e,ee)*Xy(ee));


);


*
* get statistics
*
parameter bbar(e) "Averaged estimates";
bbar(e) = sum(s, sb(s,e)) / card(s);

parameter sehat(e) "Standard errors of bootstrap algorithm";
sehat(e) = sqrt(sum(s, sqr(sb(s,e)-bbar(e)))/(card(s)-1));
```

```
parameter tbootstrap(e) "t statistic for bootstrap";
tbootstrap(e) = b(e)/sehat(e);

scalar dfbootstrap 'degrees of freedom';
dfbootstrap = card(i) - (card(e) - 1) - 1;
parameter pbootstrap(e) "p-values for bootstrap";

*
*  pvalue = 2 * pt( abs(tvalue), df)
*         = 2 * 0.5 * pbeta( df / (df + sqr(abs(tvalue))), df/2, 0.5)
*         = betareg( df / (df+sqr(tvalue)), df/2, 0.5)
*
pbootstrap(e) = betareg( dfbootstrap / (dfbootstrap+sqr(tbootstrap(e))), dfbootstrap/2, 0.5);

parameter bootstrap(e,*);
bootstrap(e,'estimates') = b(e);
bootstrap(e,'std.error') = sehat(e);
bootstrap(e,'t value') = tbootstrap(e);
bootstrap(e,'p value') = pbootstrap(e);


display
  "------------------------------------ BOOTSTRAP MODEL ------------------------",
  bootstrap;
```

The idea of this model is to build a regression equation:

$$\theta_i = \beta_0 + \beta_1 \text{BEDS}_i + \beta_2 \text{FORPROF}_i + \beta_3 \text{TEACH}_i + \beta_4 \text{RES}_i + \varepsilon_i \tag{68}$$

where θ_i are the DEA efficiency scores. From the results

```
----     290 ------------------------------------ OLS MODEL ------------------------

----     290 PARAMETER ols

           estimates   std.error     t value     p value

BEDS     1.040019E-4 1.244050E-4       0.836       0.405
FORPROF        0.099       0.042       2.390       0.019
TEACH         -0.057       0.039      -1.451       0.150
RES           -0.001 3.303407E-4      -3.133       0.002
CONST          0.607       0.035      17.330 3.59753E-31
```

we see that FORPROF is significant at $\alpha = 0.05$ (the corresponding p value is smaller than 0.05). However when we apply the resampling technique from the bootstrap algorithm, the results indicate a different interpretation:

```
----     380 ------------------------------------ BOOTSTRAP MODEL ------------------------

----     380 PARAMETER bootstrap

           estimates   std.error     t value     p value

BEDS     1.040019E-4 1.107967E-4       0.939       0.350
FORPROF        0.099       0.060       1.651       0.102
TEACH         -0.057       0.036      -1.584       0.116
RES           -0.001 2.442416E-4      -4.237 5.234667E-5
CONST          0.607       0.042      14.417 1.18732E-25
```

Here the p-value for FORPROF is indicating this parameter is *not* significant at the 0.05 level. The p-values are calculated using the incomplete beta function.

It is noted that the option `m.solvelink=2;` is quite effective for this model. Timings that illustrate this are reported in table 4.

A further small performance improvement can be achieved to augment the model equations for the DEA model by the equations that calculate $(X^T X)^{-1}$. This will

default		**solvelink=2**	
`real`	`27m12.745s`	`real`	`14m29.518s`
`user`	`20m58.595s`	`user`	`12m58.734s`
`sys`	`5m30.054s`	`sys`	`1m3.559s`

Table 4. Solvelink results

combine the DEA and OLS model into one model. After this has been done there is only one solve for each bootstrap sample.

References

1. *Gnuplot homepage*, `http://www.gnuplot.info`.
2. M. Abramovitz and I.A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, National Bureau of Standards, U.S. Govt. Printing Office, Wash., DC, 1964.
3. Robert J. Boik and James F. Robison-Cox, *Derivatives of the Incomplete Beta Function*, Tech. report, Department of Mathematical Sciences, Montana State University – Bozeman, March 1998.
4. Barry W. Brown and Lawrence B. Levy, *Certification of Algorithm 708: Significant Digit Computation of the Incomplete Beta*, ACM Transactions on Mathematical Software **20** (1994), no. 3, 393–397.
5. A. Charnes, W. W. Cooper, and E. Rhodes, *Measuring the efficiency of decision making units*, European Journal of Operational Research **2** (1978), 429–444.
6. A. Charnes, W. W. Cooper, and E. Rhodes, *Evaluating program and managerial efficiency: An application of data envelopment analysis to program follow through*, Management Science **27** (1981), 668–697.
7. M. Aslam Chaudhry and Syed M. Zubair, *On a Class of Incomplete Gamma Functions with Applications*, CRC Press, 2001.
8. Chung-Ho Chen and Chao-Yu Chou, *Determining the Optimum Process Mean under a Beta Distribution*, Journal of the Chinese Institute of Industrial Engineers **18** (2003), no. 3, 27–32.
9. W. J. Cody, *Specfun*, `www.netlib.org/specfun`.
10. ———, *An Overview of Software Development for Special Functions*, Lecture Notes in Mathematics, vol. 506, Springer Verlag, Berlin, 1976.
11. H. R. Cook, M. G. Cox, M.P. Dainton, and P.M. Harris, *Testing spreadsheets and other packages used in metrology: Testing the intrinsic functions of excel*, NPL Report CISE 27/99, Report to the National Measurement System Policy Unit, Department of Trade and Industry, September 1999.
12. Microsoft Corp., *Microsoft Knowledge Base Article - 215214*, `http://support.microsoft.com/`.
13. D. R. Cox and E. J. Snell, *Applied Statistics: Principles and Examples*, Chapman and Hall, London, 1981.
14. B. Decaluwé, A. Patry, L. Savard, and E.Thorbecke, *Poverty Analysis Within a General Equilibrium Framework*, Working Paper 9909, CRÉFA, 1999.
15. Bernard Decaluwé, André Patry, and Luc Savard, *Income Distribution, Poverty Measures and Trade Shocks: A Computable General Equilibrium Model of a Archetype Developing Country*, Cahier de recherche 9812, Département d'économique, Université Laval, 1998.
16. Armido R. Didonato and Alfred H. Morris Jr., *Algorithm 708: Significant Digit Computation of the Incomplete Beta Function Ratios*, ACM Transactions on Mathematical Software **18** (1992), no. 3, 360–373.
17. Bradley Efron and Robert J. Tibshirani, *An Introduction to the Bootstrap*, Chapman & Hall, 1993.
18. Ali Emrouznejad, *Dea homepage*, `http://www.deazone.com/`, 2001.
19. Merran Evans, Nicholas Hastings, and Brian Peacock, *Statistical Distributions*, 3rd ed., Wiley, 2000.

20. Brian J. Francis, *Remark AS R88: A Remark on Algorithm AS 121: The Trigamma Function*, Applied Statistics **40** (1991), no. 3, 514–515.
21. J. W. Glaisher, *On a Class of Definite Integrals*, Philosophical Magazine **XXXII** (1871), 294–301.
22. Student (W. S. Gosset), *The probable error of a mean*, Biometrika **6** (1908), no. 1, 1–25.
23. W. E. Griffiths, R. C. Hill, and G. G. Judge, *Learning and Practicing Econometrics*, Wiley, 1993.
24. Julian Havil, *Gamma : Exploring Euler's Constant*, Princeton University Press, 1993.
25. Norman L. Johnson, Samuel Kotz, and N. Balakrishnan, *Continuous Univariate Distributions, Volume 1*, 2nd ed., Wiley, 1994.
26. Erwin Kalvelagen, *Efficiently Solving DEA Models with GAMS*, `http://amsterdamoptimization.com/pdf/dea.pdf`.
27. L. Knüsel, *On the accuracy of statistical distributions in Microsoft Excel 97*, Computational Statistics and Data Analysis **26** (1998), 375–377.
28. ———, *On the Reliability of Microsoft Excel XP for Statistical Purposes*, Tech. report, University of Munich, Department of Statistics, undated.
29. George Marsaglia, *Evaluating the Normal Distribution*, Journal of Statistical Software **11** (2004).
30. B. D. McCullough and B. Wilson, *On the accuracy of statistical procedures in Microsoft Excel 97*, Computational Statistics and Data Analysis **31** (1999), 27–37.
31. R. J. Moore, *Algorithm AS 187: Derivatives of the Incomplete Gamma Integral*, Applied Statistics **31** (1982), no. 3, 330–335.
32. Akihiro Ogura, *Post-Gaussian Variational Method for Quantum Anharmonic Oscillator*, Tech. report, Laboratory of Physics, College of Science and Technology, Nihon University, 1999.
33. M. D. Phillips and B.-R. Cho, *A Nonlinear Model for Determining the Most Economic Process Mean under a Beta Distribution*, International Journal of Reliability, Quality and Safety Engineering **7** (2000), 61–74.
34. William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, *Numerical Recipes in Fortran*, 2nd ed., Cambridge University Press, 1992.
35. R Development Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2004, ISBN 3-900051-00-3.
36. B.E. Schneider, *Algorithm AS 121: Trigamma Function*, Applied Statistics **27** (1978), no. 1, 97–99.
37. B. Shea, *Algorithm AS 239: Chi-squared and Incomplete Gamma Integral*, Applied Statistics **37** (1988), no. 3, 466–473.
38. Leopold Simar and Paul W. Wilson, *Sensitivity Analysis of Efficiency Scores: How to Bootstrap in Nonparametric Frontier Models*, Journal of Applied Statistics **44** (1998), no. 1, 49–61.
39. ———, *A general methodology for bootstrapping in nonparametric frontier models*, Journal of Applied Statistics **27** (2000), 779–802.
40. N. J. A Sloane, *The On-Line Encyclopedia of Integer Sequences; Sequence A030169*, `http://www.research.att.com/projects/OEIS?Anum=A030169`.
41. ———, *The On-Line Encyclopedia of Integer Sequences; Sequences A001163,A001164*, `http://www.research.att.com/~njas/sequences/Seis.html`.
42. Jerome Spanier and Keith B. Oldham, *An atlas of functions*, Hemisphere, 1987.
43. Luke Tierney, *XLISP-STAT, A Statistical Environment Based on the XLISP Language (Version 2.0)*, Tech. Report 528, University of Minnesota, School of Statistics, July 1989.
44. Eric W. Weisstein, *Beta Function*, From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/BetaFunction.html`.
45. ———, *Erf*, From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/Erf.html`.
46. ———, *Gamma Function*, From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/GammaFunction.html`.
47. Robert L. Wolpert, *The Gamma Function*, Tech. report, Institute of Statistics and Decision Sciences, Duke University, Durham, NC, August 2000.
48. Mei Xue and Patrick T. Harker, *Overcoming the Inherent Dependency of DEA Efficiency Scores: A Bootstrap Approach*, Tech. report, Department of Operations and Information Management, The Wharton School, University of Pennsylvania, April 1999.

Amsterdam Optimization Modeling Group LLC, Washington D.C./The Hague
E-mail address: erwin@amsterdamoptimization.com