
Subject: Re: Random # generator?

Posted by [meron](#) on Tue, 30 Jan 1996 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

In article <4em2j8\$ga8@ra.nrl.navy.mil>, nicholas@uap.nrl.navy.mil (Andy Nicholas) writes:

> Hi,

> I'm looking for a random number generator that uses Poisson

> statistics rather than Gaussian (that is what the IDL routine uses).

> I'm modeling some low-count rate noise in Dynamics Explorer images.

> Anybody out there have one, or know where I could find one? Thought I'd

> check before i re-invent the wheel....

> Thanks,

> -Andy

>

The following function should do what you want. It takes as an input an arbitrary array and adds a Poisson noise to it. Thus, if you call it with an array initialized to a given constant it'll return an array with Poisson distributed random numbers (with the constant being the mean). The function calls two other functions, CAST and TYPE which are provided underneath.

Function Noise_Poiss, dat

```
on_error, 1
```

```
neld = n_elements(dat)
```

```
if neld eq 0 then message, 'No data!'
```

```
wdat = Cast(dat,2,3) > 0
```

```
res = -wdat
```

```
u = where(res lt 0, ndo)
```

```
if ndo gt 0 then begin
```

```
ddat = 2*wdat
```

```
sdat = sqrt(ddat)
```

```
mdat = sdat/(ddat + 1./6)
```

```
edat = fltarr(neld)
```

```
edat(u) = 1 + alog(wdat(u))
```

```
while ndo gt 0 do begin
```

```
    kres = wdat(u) + sdat(u)*tan(!pi*(randomu(se,ndo) - 0.5))
```

```
    kres = Int(-1e9 > kres < 1e9)
```

```
    v = where(kres ge 0, ndo)
```

```
    if ndo gt 0 then begin
```

```
        u = u(v)
```

```
        kpr = kres(v) + 0.5
```

```
        rat = exp(kpr*(edat(u) - alog(kpr - (1./24)/kpr)) - wdat(u))/ $
```

```
            atan(sdat(u), (kpr - wdat(u))^2 + ddat(u) - 0.25)*mdat(u)
```

```

w = where(rat ge randomu(se,ndo), ndo)
if ndo gt 0 then res(u(w)) = kres(v(w))
  endif
  u = where(res lt 0, ndo)
endwhile
endif

return, res
end

```

Function Cast, x, low, high

```

;+
; NAME:
; CAST
; PURPOSE:
; Generalized type casting. Converts all variables whose type code is
; out of the range [LOW,HIGH] into this range.
; CATEGORY:
; Type conversion
; CALLING SEQUENCE:
; Result = CAST( X, [LOW [,HIGH]])
; INPUTS:
;   X
; Numerical, arbitrary, or a character representation of a number(s).
;   LOW
; Number representing a type code, range (1:9). If greater than 9, it is
; set to 9. If less than 1, or not given, it is set to 1.
; OPTIONAL INPUT PARAMETERS:
;   HIGH
; Type code, same as LOW. Default value is 9. If provided and less than
; LOW, it is set to LOW.
; KEYWORD PARAMETERS:
; None.
; OUTPUTS:
; If the type of X is < LOW, CAST returns X converted to type LOW.
; If the type of X is > HIGH, CAST returns X converted to type HIGH.
; Otherwise CAST returns X.
; OPTIONAL OUTPUT PARAMETERS:
; None.
; COMMON BLOCKS:
; None.
; SIDE EFFECTS:
; None.
; RESTRICTIONS:

```

```

; 1) An attempt to convert a string which is NOT a character
; representation of a number into a numeric type will yield error.
; 2) X cannot be a structure (but can be a structure element).
; 3) The value 8 for either LOW or HIGH is not allowed (since it
; corresponds to structure type).
; PROCEDURE:
; Identifies the type of X, and if out of the range given by [LOW,HIGH]
; calls the proper conversion routine using the system routine
; CALL_FUNCTION. Also uses TYPE from MIDL.
; MODIFICATION HISTORY:
; Created 25-DEC-1991 by Mati Meron.
; Modified 15-JUN-1995 by Mati Meron to accept the new DOUBLECOMPLEX type.
;-

```

```

on_error, 1
conv = ['nada', 'byte', 'fix', 'long', 'float', 'double', 'complex', $
'string', 'nonap', 'dcomplex']
if n_elements(low) eq 0 then ilo = 1 else ilo = 1 > fix(low) < 9
if n_elements(high) eq 0 then ihi = 9 else ihi = ilo > fix(high) < 9

ityp = Type(x)
if ilo eq 8 or ihi eq 8 or ityp eq 8 or ityp eq 0 then $
message, 'Can''t do that!' else $
if ityp lt ilo then return, call_function(conv(ilo),x) else $
if ityp gt ihi then return, call_function(conv(ihi),x) else return, x

end

```

Function Type, x

```

;+
; NAME:
; TYPE
; PURPOSE:
; Finds the type class of a variable.
; CATEGORY:
; Programming.
; CALLING SEQUENCE:
; Result = TYPE(X)
; INPUTS:
; X
; Arbitrary, doesn't even need to be defined.
; OPTIONAL INPUT PARAMETERS:
; None.
; KEYWORD PARAMETERS:
; None.

```

```
; OUTPUTS:
; Returns the type of X as a long integer, in the (0,9) range.
; OPTIONAL OUTPUT PARAMETERS:
; None.
; COMMON BLOCKS:
; None.
; SIDE EFFECTS:
; None.
; RESTRICTIONS:
; None.
; PROCEDURE:
; Extracts information from the SIZE function.
; MODIFICATION HISTORY:
; Created 15-JUL-1991 by Mati Meron.
;-
```

```
    dum = size(x)
    return, dum(dum(0) + 1)
end
```

Mati Meron | "When you argue with a fool,
meron@cars3.uchicago.edu | chances are he is doing just the same"

Subject: Re: Random # generator?
Posted by [landsman](#) on Tue, 30 Jan 1996 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <4em2j8\$ga8@ra.nrl.navy.mil>, nicholas@uap.nrl.navy.mil (Andy Nicholas) writes...
> Hi,
> I'm looking for a random number generator that uses Poisson
> statistics rather than Gaussian (that is what the IDL routine uses).

The procedure POIDEV in the IDL Astronomy Library should do what you want.
The Library is located at <http://idlastro.gsfc.nasa.gov/homepage.html>

--Wayne Landsman landsman@sorbet.gsfc.nasa.gov
