A Numerical Issue in Computing the Rician and Non-central Chi-Square Probability Density Functions

Consider a random variable $x = m + z$, where m is a (possibly complex) constant and z is complex noise with i.i.d. zero-mean Gaussian real and imaginary parts of variance $\sigma_z^2/2$; thus the total noise variance is σ_z^2. Let $X = |x|$. The probability density function (pdf) of X is the Rician distribution [1]

$$p(X) = \begin{cases} \dfrac{2X}{\sigma_z^2}\exp\left[-\dfrac{1}{\sigma_z^2}\left(X^2+m^2\right)\right] I_0\left(\dfrac{2mX}{\sigma_z^2}\right), & X > 0 \\ 0, & X < 0 \end{cases} \tag{1}$$

The pdf of $Y = X^2$ is a non-central chi-square distribution [1]

$$p(Y) = \begin{cases} \dfrac{1}{\sigma_z^2}\exp\left[-\dfrac{1}{\sigma_z^2}\left(Y+m^2\right)\right] I_0\left(\dfrac{2\sqrt{m^2Y}}{\sigma_z^2}\right), & Y > 0 \\ 0, & Y < 0 \end{cases} \tag{2}$$

Both the Rician and non-central chi-square pdfs involve the product of a decaying exponential and a zero$^{\text{th}}$-order modified Bessel function $I_0(x)$. MATLAB provides a modified Bessel function call, `besseli(order,argument)`. For instance, Eqn. (1) can be expressed in MATLAB as

```
b = m;
c = sqrt(sigz^2/2);
p(X) = (X/c^2).*exp(-(X.^2+b^2)/2/c^2).*besseli(0,X*b/c^2);
```

Unfortunately, the computation of $I_0(X)$ overflows in `besseli` for $X > 700$. This occurs because, for large X, $I_0(X)$ is proportional to e^x. The Rician or non-central chi-square are themselves well-behaved, because they are the product of this growing exponential with a more rapidly decaying (quadratic *vs.* linear argument) exponential. Nonetheless, since the two terms are computed separately in the line above, the computation overflows.

This problem can be solved by using a large-argument approximation to $I_0(X)$. Specifically [2],

$$I_0(X) \approx \frac{1}{\sqrt{2\pi X}} e^{-X}\left\{1 - \frac{1}{8X} + \frac{9}{2!(8X)^2} - \frac{225}{3!(8X)^3} + \dots\right\} \tag{3}$$

For $X > 12.5$, the second term in the sum is less than 1% of the first term; for $X > 125$ it is less than 0.1%. At the same time, this is well within the range of arguments for which the `besseli(0,X)` function converges. Thus we can use `besseli()` for *X* less than 100 or so, and for *x* greater than 100, Eqn. (3) simplifies to simply

$$I_0(X) \approx \frac{1}{\sqrt{2\pi X}} e^{-X}, \quad x > 100 \tag{4}$$

Using this in (1) gives the better-behaved expression

$$p(X) = \begin{cases} \sqrt{\dfrac{X}{\pi m \sigma_z^2}} \exp\left[-\dfrac{1}{\sigma_z^2} (X-m)^2 \right], & X > 0 \\ 0, & X < 0 \end{cases} \tag{5}$$

A MATLAB function called `rician` was written to compute the Rician pdf. It uses (1) for arguments less than 200, and (5) for arguments greater than 200. A listing is included at the end of this memorandum.

The same issue occurs in computation of the non-central chi-square distribution of (2). Using the approximation (4) again gives

$$p(Y) = \begin{cases} \dfrac{2}{\sigma_z^2 \sqrt{8\pi m \sqrt{Y}}} \exp\left[-\dfrac{1}{\sigma_z^2} \left(\sqrt{Y} - m\right)^2 \right], & Y > 0 \\ 0, & Y < 0 \end{cases} \tag{6}$$

Again, a MATLAB function called `noncentralchisquare` was written to compute the non-central chi-square pdf. It uses (2) for arguments less than 200, and (6) for arguments greater than 200. A listing is included at the end of this memorandum.

Reference

[1] S. M. Kay, *Fundamentals of Statistical Signal processing, vol. II: Detection Theory*. Prentice-Hall, New York, 1998.

[2] M. Abramowitz and I. A. Stegun, editors, *Handbook of Mathematical Functions* (10th printing, Dec. 1972). U. S. Dept. of Commerce, Natl. Bureau of Standards, Applied Mathematics Series – 55, June 1964.Section 9.7.1, p. 377.

Listing of `rician.m`

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% function rician(a,b,c)
%
% function defined to compute Rician for large arguments, where MATLAB's
% besseli(0,x) overflows.  Uses approximation that, for large x, I0(x) ~=
% (1/sqrt(2*pi*x))*exp(x).
%
%  Specifically, we compute the function
%  p(a) = (a/c^2)*exp(-(a.^2+b^2)/2/c^2)*I0(a*b/c^2)
%
%  'a' is a vector ranging over the range of interest
%  'b' and 'c' are constants
%
% M. A. Richards, May 2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function p = rician(a,b,c)

p = zeros(size(a));
kb = find(2*a*b/c^2<200);  % besseli(0,x) doesn't actually blow until x > 700
ka = find(2*a*b/c^2>=200);

p(kb) = (a(kb)/c^2).*exp(-(a(kb).^2+b^2)/2/c^2).*besseli(0,a(kb)*b/c^2);
p(ka) = sqrt(a(ka)/(2*pi*b*c^2)).*exp(-(a(ka)-b).^2/2/c^2);

end
```

Listing of `noncentralchisquare.m`

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% function noncentralchisquare(a,b,c)
%
% function defined to compute non-central chi-square density for large
% arguments, where MATLAB's besseli(0,x) overflows.  Uses approximation
% that, for large x, I0(x) ~= (1/sqrt(2*pi*x))*exp(x).
%
%  Specifically, we compute the function
%  p(a) = (1/2/c^2)*exp(-(a+b^2)/2/c^2)*I0(sqrt(a)*b/c^2)
%
%  'a' is a vector ranging over the range of interest
%  'b' and 'c' are constants
%
% M. A. Richards, May 2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function p = noncentralchisquare(a,b,c)

p = zeros(size(a));
kb = find(sqrt(a)*b/c^2<200);  % besseli(0,x) doesn't actually blow until x >
700
ka = find(sqrt(a)*b/c^2>=200);

p(kb) = (1/2/c^2)*exp(-(a(kb)+b^2)/2/c^2).*besseli(0,sqrt(a(kb))*b/c^2);
p(ka) = (1/c./sqrt(8*pi*b*sqrt(a(ka)))).*exp(-(sqrt(a(ka))-b).^2/2/c^2);

end
```