Mathematical Statistics with *Mathematica*

Colin Rose and Murray D. Smith

Abstract

We present a unified approach for doing mathematical statistics with *Mathematica*. At one extreme, our package **PDF** empowers even the "statistically challenged" with the ability to perform complicated operations without realizing it. At the other extreme, it enables the professional statistician to tackle tricky multivariate distributions, generating functions, transformations, symbolic maximum likelihood estimation, unbiased estimation, checking (and correcting) of textbook formulae, and so on. By taking full advantage of the latest v4 Assumptions technology, the **PDF** package can produce exceptionally clean and neat symbolic output.

Dr Colin Rose — *Theoretical Research Institute*, Sydney — colin@tri.org.au
Dr Murray D. Smith — *University of Sydney* — Murray.Smith@econ.usyd.edu.au

1 Introduction

1.1 A New Approach

The use of computer software in statistics is far from knew. Indeed, literally hundreds of statistical computer programs exist. Yet, underlying existing programs is almost always a numerical / graphical view of the world. *Mathematica* can easily handle the numerical and graphical sides, but it offers in addition an extremely powerful and flexible symbolic computer algebra system. The **PDF** software package builds upon that symbolic engine to create a sophisticated toolset specially designed for doing mathematical statistics. This forms the basis of a Springer-Verlag text entitled *Mathematical Statistics with Mathematica* (500 pages) that ships with CD-ROM, custom palettes, online help, and live interactive chapters. This provides a simple and unified approach for doing mathematical statistics with *Mathematica*, suited to those in statistics, econometrics, engineering, physics, psychometrics, economics, finance, biometrics - indeed across the full ambit of the physical/social sciences and across both the pure and the applied domains. It is expected to be available by January 2000.

■ 1.2 Features

Features include: * a complete suite of functions for manipulating probability density functions, * symbolic maximum likelihood estimation, * numerical maximum likelihood estimation, * automated symbolic Pearson curve fitting, * Johnson fitting, * Gram-Charlier expansions, * nonparametric kernel density estimation, * moment conversion formulae (convert automatically between cumulants, raw moments, central moments, and factorial moments: univariate and multivariate), * random number generation for any discrete distribution, * fully automated transformations (functions of random variables), * asymptotics, * decision theory (order statistics, mean square error, ...), * unbiased estimation (h-statistics, k-statistics, polykays). The **PDF** Application Package can replace many reference works and extends the analysis to problems of arbitrary high order.

■ 1.3 Design Philosophy

The **PDF** package has been designed with two central goals: it sets out to be **general**, and it strives to be **'delightfully simple'**.

By **general**, we mean that it should *not* be limited to a set of special or well-known textbook distributions. It should *not* operate like a textbook appendix with prepared 'crib sheet' answers. Rather, it should know how to solve problems from first principles. It should seamlessly handle: univariate *and* multivariate distributions, continuous *and* discrete random variables, distributions of functions of random variables, all with *and* without parameters – and distributions no-one has ever thought of before.

By **delightfully simple**, we mean both (i) easy to use, and (ii) able to solve problems that seem difficult but which are formally quite simple. Consider, for instance, playing a devilish game of chess against a strong chess computer: in the middle of the game, after a short pause, the computer blandly announces "Mate in 16 moves". The problem it has solved might seem fantastically difficult, but it is really just a 'delightfully simple' finite problem that is conceptually no different to looking just 2 moves ahead. Of course, as soon as one has a tool for solving such problems, the notion of what is 'difficult' changes completely. A pocket calculator is certainly a delightfully simple device: it is easy to use, and it can solve tricky problems that were previously thought to be difficult. But today, few people bother to ponder at the marvel of a calculator any more, and we now generally spend our time either using such tools, or trying to solve higher-order conceptual problems ... and so, we are certain, it will be with mathematical statistics too. In fact, while much of the material traditionally studied in mathematical statistics courses may appear to many to be difficult, such material is often really just delightfully simple. Normally, all we want is an expectation, or a probability, or a transformation. But once we are armed with a computerised expectation operator, we can find any kind of expectation including the mean, variance, skewness, kurtosis, moment generating function, characteristic function, raw moments, central moments, cumulants, probability generating function, factorial moment generating function, and so on. Normally, many of these calculations are not attempted in undergraduate texts, because the mechanics are deemed too hard. Any yet, underlying all of them is just the delightfully simple expectation operator.

2 Core Functions

2.1 Getting started

The **PDF** package adds over a hundred new functions to *Mathematica*. But 95% of the time, we can get by with just 4 of them:

abbreviation	*description*	
`PlotDensity[ f ]`	Plotting (automated)	
`Expect[x, f ]`	Expectation operator	$E[X]$
`Prob[x, f]`	Probability	$P(X \le x)$
`Transform[eqn, f ]`	Transformations	

Table 1: Core functions for a random variable X with density $f(x)$

This ability to handle plotting, expectations, probability, and transformations, with just 4 functions, makes the **PDF** system very easy to use, even for those not familiar with *Mathematica.*

To illustrate, let us suppose the continuous random variable X has probability density function (pdf) $f(x) = \frac{1}{\pi \sqrt{1-x} \sqrt{x}}$, where $x \in (0, 1)$. In *Mathematica*, we enter this as:

```
f = 1/(π √(1 - x) √x);        domain[f] = {x, 0, 1};
```

This is known as the Arc-Sine distribution. Here is a plot of $f(x)$:

```
PlotDensity[f];
```

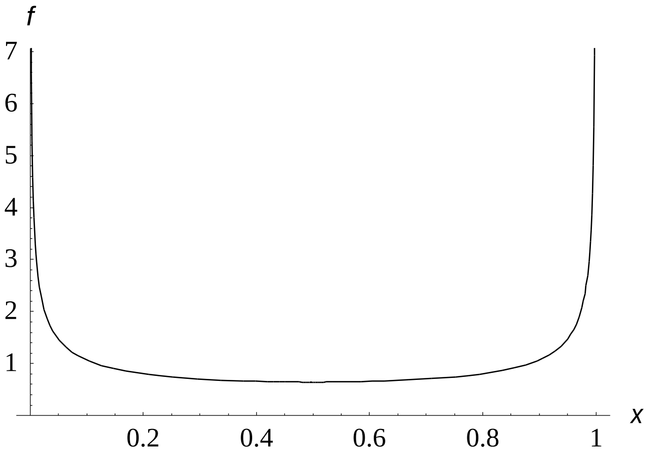

Fig. 1: The Arc-Sine pdf

Here is the cumulative distribution function (cdf), $P(X \leq x)$, which also provides the clue to the naming of this distribution:

```
Prob[x, f]
```

$$\frac{2\,\text{ArcSin}[\sqrt{x}]}{\pi}$$

The mean, $E[X]$, is:

```
Expect[x, f]
```

$$\frac{1}{2}$$

while the variance of X is:

```
Var[x, f]
```

$$\frac{1}{8}$$

The r^{th} moment of X is $E[X^r]$:

```
Expect[x^r, f]
```

$$\frac{\Gamma[\frac{1}{2}+r]}{\sqrt{\pi}\ \Gamma[1+r]}$$

The moment generating function (mgf) of X is $E[e^{tX}]$:

```
Expect[ℯ^(t x), f]
```

$$e^{t/2}\ \text{BesselI}[0, \frac{t}{2}]$$

Now consider the transformation to a new random variable Y such that $Y = \sqrt{X}$. By using the `Transform` and `TransformExtremum` functions, the pdf of Y, say $g(y)$, and the domain of its support can be found:

```
        g = Transform[y == √x, f]
domain[g] = TransformExtremum[y == √x, f]
```

$$\frac{2\,y}{\pi\sqrt{y^2 - y^4}}$$

```
{y, 0, 1}
```

So, we have started out with a quite arbitrary pdf $f(x)$, transformed it to a new one $g(y)$, and since both density g and its `domain` have been inputted into *Mathematica*, we can also apply the **PDF** tool set to density g.

■ 2.2 Working with parameters (Assumptions technology)

The **PDF** package has been designed to seamlessly support parameters. It does so by taking full advantage of the new Assumptions technology introduced in Version 4 of *Mathematica*. To illustrate, let us consider the familiar Normal distribution with mean m and variance s^2. That is, let $X \sim N(m, \; s^2)$, where m Œ — and s > 0. We enter the pdf $f(x)$ in the standard way, but this time we have some extra information about the parameters m and s which can be added to the end of the `domain[f]` statement:

$$f = \frac{1}{s\sqrt{2\,p}}\,\text{Exp}\left[-\frac{(x-m)^2}{2\,s^2}\right];$$

```
domain[f] = {x, -∞, ∞} && {m Œ Reals, s > 0};
```

From now on, the assumptions about m and s will be 'attached' to density *f*, so that whenever we operate on density *f* with a **PDF** package function, these assumptions will be automatically applied in the background. With this new technology, the **PDF** package can usually produce remarkably crisp textbook-style answers, even when working with very complicated distributions.

The **PDF** package function, `PlotDensity`, makes it extremely easy to examine the effect of changing parameter values. For instance, here is the pdf plotted at three different values of s:

```
PlotDensity[f /. {m Æ 0, s Æ {1, 2, 3}}];
```

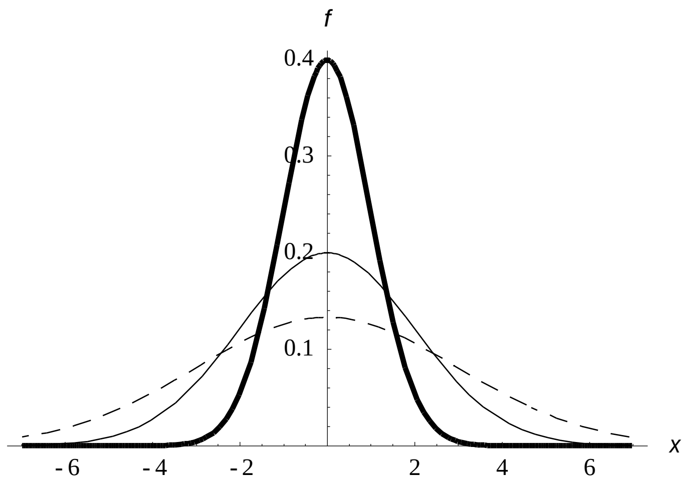

Fig. 2: The pdf of a Normal random variable, when m = 0 and s = 1, 2 and 3

It is well known that $E[X] = $ m and $\text{Var}(X) = s^2$, as we can easily verify:

```
Expect[x, f]
```

m

```
Var[x, f]
```

s^2

Because the **PDF** package is completely general in its design, we can just as easily solve problems that are both less well-known and more 'difficult', such as finding Var(X^2) :

```
Var[x², f]
```

$2\ (2\ m^2\ s^2 + s^4)$

Assumptions technology is a very important addition to *Mathematica*. In order for it to work, one should enter as much information about parameters as possible: not only will the resulting answer be much neater, it may also be obtained faster, and it may make it possible to solve problems that could not otherwise be solved. Here are some examples of Assumptions statements:

```
{a > 1,  b Œ Integers,  -∞ < g < p,  d Œ Reals,  q > 0}
```

The **PDF** package implements assumptions technology in a *distribution*-specific manner. That is, the assumptions are attached to the density $f(x;\ q)$, and not to the parameter q. Because the assumptions are not global, it is important to realise that the assumptions will only be automatically invoked when using the suite of **PDF** package functions. By contrast, *Mathematica*'s inbuilt functions such as the derivative function, `D[f, x]`, will not automatically assume that $q > 0$ when presenting their solution.

■ 2.3 Discrete random variables

The **PDF** toolset automatically handles discrete random variables in the same way. The only difference is that, when we define the density, we add a flag to tell *Mathematica* that the random variable is `{Discrete}`. For instance:

Let the discrete random variable X have probability mass function (pmf) $f(x) = \binom{r+x-1}{x} p^r (1-p)^x$, where $x = 0, 1, 2, ...$ Here parameter p is the probability of success, while parameter r is a positive integer. In *Mathematica*, we enter this as:

```
      f = Binomial[r + x - 1, x] p^r (1 - p)^x;
domain[f] = {x, 0, ∞} && {Discrete} &&
                       {0 < p < 1,  r > 0,  r Œ Integers} ;
```

This is known as the Pascal distribution. Here is a plot of $f(x)$:

```
PlotDensity[f /. {p Æ 1/2, r Æ 10 }];
```

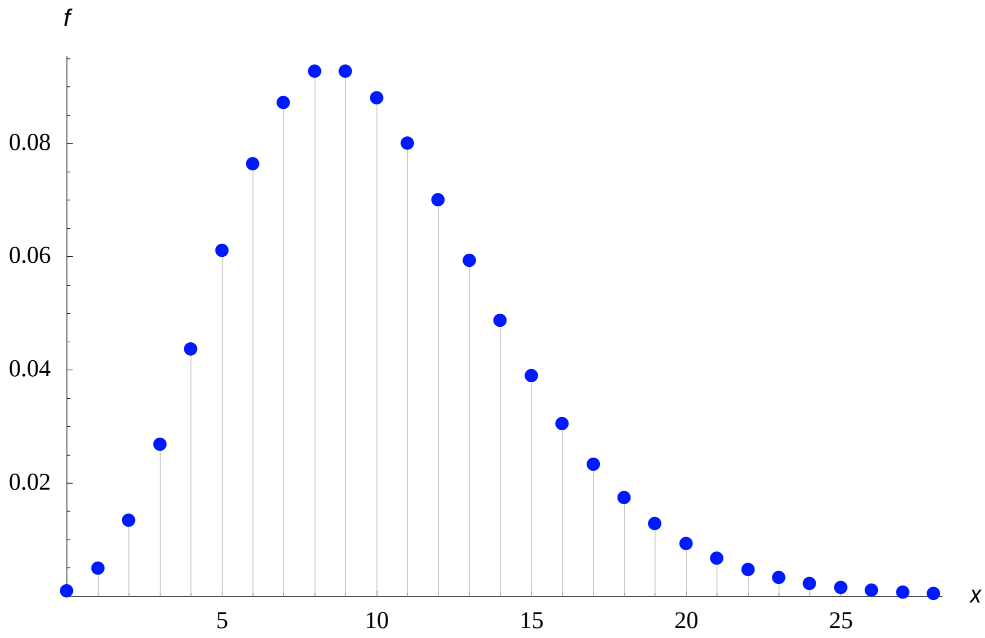

Fig. 3: The pmf of a Pascal discrete random variable

Here, for example, is the cdf, equal to $P(X$ £ $x)$,

```
Prob[x, f]
```

```
1 - ((1 - p)^(1+Floor[x]) p^r G[1 + r + Floor[x]] Hypergeometric2F1[1,
     1 + r + Floor[x], 2 + Floor[x], 1 - p]) / (G[r] G[2 + Floor[x]])
```

The mean $E[X]$ and variance of X are given by:

```
Expect[x, f]
```

$$\left(-1 + \frac{1}{p}\right) r$$

```
Var[x, f]
```

$$\frac{r - p\,r}{p^2}$$

The probability generating function (pgf) is $E\,[t^X]$:

```
Expect[t^x, f]
```

$$p^r\,(1 + (-1 + p)\,t)^{-r}$$

■ 2.4 Multivariate pdf's

The **PDF** toolset extends naturally to a multivariate setting. To illustrate, let us suppose that X and Y have joint pdf $f(x, y)$ with support $x > 0$, $y > 0$:

```
f = „^(-2 (x+y)) („^(x+y) + (-2 + „^x) (-2 + „^y) a);
domain[f] = {{x, 0, ∞}, {y, 0, ∞}}  &&  {-1 < a < 1};
```

where parameter a is such that $-1 < a < 1$. This is known as Morgenstern's (1956) Bivariate Exponential pdf. Here is a plot of $f(x, y)$:

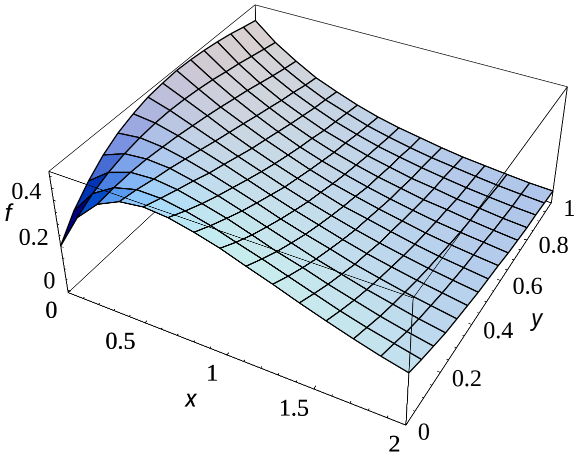

Fig. 4: Morgenstern's Bivariate Exponential pdf when $a = -0.8$

Here is the cdf, namely $P(X \leq x, Y \leq y)$,

```
Prob[{x, y}, f]
```

```
„^(-2 (x+y)) (-1 + „^x) (-1 + „^y) („^(x+y) + a)
```

Here is Cov(X, Y), the covariance between X and Y:

```
Cov[{x, y}, f]
```

$\frac{a}{4}$

More generally, here is the variance-covariance matrix:

```
Varcov[f] // MatrixForm
```

$$\begin{pmatrix} 1 & \frac{a}{4} \\ \frac{a}{4} & 1 \end{pmatrix}$$

Here is the marginal density of X:

```
Marginal[x, f]
```

e^{-x}

Here is the conditional pdf of Y, given $X = x$:

```
Conditional[x, f]
```

Here is the conditional pdf f(y | x):

$e^{x-2(x+y)} (e^{x+y} + (-2 + e^{x}) (-2 + e^{y}) a)$

Here is the bivariate mgf $E[e^{t_1 X + t_2 Y}]$:

```
mgf = Expect[e^(t1 x + t2 y), f]
```

$$\frac{4 - 2 t_2 + t_1 (-2 + (1 + a) t_2)}{(-2 + t_1) (-1 + t_1) (-2 + t_2) (-1 + t_2)}$$

Differentiating the mgf is one way to derive moments. Here is the product moment $E[X^2 Y^2]$:

```
D[mgf, {t1, 2}, {t2, 2}] /. t_ → 0 // Simplify
```

$4 + \frac{9 a}{4}$

which we could otherwise have found directly with:

```
Expect[x^2 y^2, f]
```

$4 + \frac{9 a}{4}$

Multivariate transformations pose no problem either. For instance, let $U = \frac{Y}{1+X}$ and $V = \frac{1}{1+X}$. Then our transformation equation is:

```
eqn = {u == y/(1 + x), v == 1/(1 + x)};
```

Using `Transform`, we can find the joint density of random variables U and V, denoted $g(u, v)$:

```
g = Transform[eqn, f]
```

$$\frac{e^{\frac{-2-2u+v}{v}} \left(4 e a - 2 e^{\frac{1}{v}} a - 2 e^{\frac{u+v}{v}} a + e^{\frac{1+u}{v}} (1 + a)\right)}{v^3}$$

while the extremities of the new random variables are:

```
TransformExtremum[eqn, f]
```

{{u, 0, ∞}, {v, 0, 1}}

3 Specialised Functions

≈ ***Example 1:*** Symbolic MLE

Although statistical software has long been used for maximum likelihood (ML) estimation, the focus of attention has almost always been on obtaining ML estimates (a *numerical* problem), rather than on deriving ML estimators (a *symbolic* problem). The **PDF** package makes it possible to derive *exact* symbolic ML estimators from first principles with a computer algebra system.

For instance, consider the following simple problem: let $(X_1, \ldots, X_n)$ denote a random sample of size n collected on $X \sim \text{Rayleigh}(s)$, where parameter $s > 0$ is unknown. We wish to find the ML estimator of s. We begin in the usual way by inputting the likelihood function into *Mathematica*:

$$\mathbf{L} = \prod_{i=1}^{n} \frac{x_i}{s^2}\, \mathbf{Exp}\left[-\frac{x_i^2}{2\, s^2}\right];$$

If we try to evaluate the log-likelihood:

Log[L]

$$\text{Log}\left[\prod_{i=1}^{n} \frac{e^{-\frac{x_i^2}{2 s^2}}\, x_i}{s^2}\right]$$

... nothing happens ! {*Mathematica* assumes nothing about the symbols that have been input, so its inaction is perfectly reasonable.} But we can enhance `Log` to do what is wanted here using the **PDF** package function `SuperLog`. To activate this enhancement, we first switch it on:

SuperLog[On]

```
SuperLog is now On.
```

If we now evaluate `Log[L]` again, we obtain a much more useful result:

logL = Log[L]

$$-2\, n\, \text{Log}[s] + \sum_{i=1}^{n} \text{Log}[x_i] - \frac{\sum_{i=1}^{n} x_i^2}{2\, s^2}$$

Then the score is:

```
score = ∂s logL
```

$$-\frac{2\,n}{s} + \frac{\sum_{i=1}^{n} x_i^2}{s^3}$$

Setting the score to zero defines the ML estimator s. The resulting equation can be easily solved using *Mathematica*'s `Solve` function. The ML estimator s is given as a replacement rule → for s:

```
s = Solve[score == 0, s]⟦2⟧
```

$$\left\{ s \rightarrow \frac{\sqrt{\sum_{i=1}^{n} x_i^2}}{\sqrt{2}\,\sqrt{n}} \right\}$$

The second order conditions (evaluated at the first order conditions) are always negative, which confirms that s is indeed the ML estimator :

```
SOC = ∂{s,2} logL /. s
```

$$-\frac{8\,n^2}{\sum_{i=1}^{n} x_i^2}$$

Finally, let us suppose that an observed random sample is {1, 0, 3, 4}:

```
data = {1, 0, 3, 4};
```

Then the ML estimate of s is obtained by substituting this data into the ML estimator s:

```
s /. {n → 4, xi :> data⟦i⟧}
```

$$\left\{ s \rightarrow \frac{\sqrt{13}}{2} \right\}$$

≈ ***Example 2:*** Pearson Fitting

Karl Pearson showed that if we know the first four moments of a distribution, we can construct a density function that is consistent with those moments. This can provide a neat way to build density functions that approximate a given set of data. For instance, for a given data set, let us suppose that:

```
mean = 37.875;
m̂234 = {191.55, 1888.36, 107703.3};
```

denoting estimates of the mean, and of the second, third and fourth central moments. The Pearson family consists of 7 main *Types*, so our first task is to find out which type this data is consistent with. We do this with the `PearsonPlot` function:

```
PearsonPlot[m̃234]
```

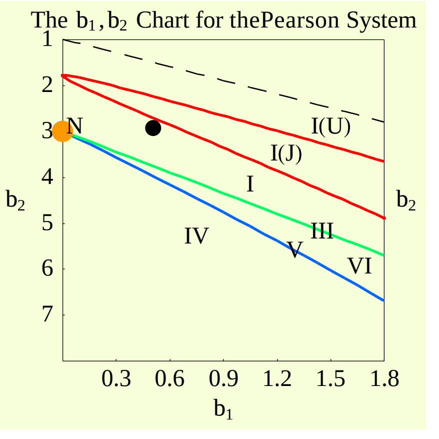

Fig. 5: The b_1, b_2 chart for the Pearson system

The big black dot is in the *Type I* zone. Then the fitted Pearson density $f(x)$ and its domain are immediately given by:

```
{f, domain[f]} = PearsonI[mean, m̃234, x]

{9.62522¥10^-8 (94.3127 - 1. x)^2.7813 (-16.8709 + 1. x)^0.407265,
 {x, 16.8709, 94.3127}}
```

It's that easy. The actual data used to create this example is quite well known grouped data, and depicts the number of sick people (`freq`) at different ages (`M`):

```
   M = {17,  22,  27,  32,  37,  42, 47, 52, 57, 62, 67, 72, 77, 82, 87};
freq = {34, 145, 156, 145, 123, 103, 86, 71, 55, 37, 21, 13,  7,  3,  1};
```

We can easily compare the 'histogram' of the empirical data with our smooth fitted Pearson pdf:

```
FrequencyGroupPlot[{M, freq}, f];
```

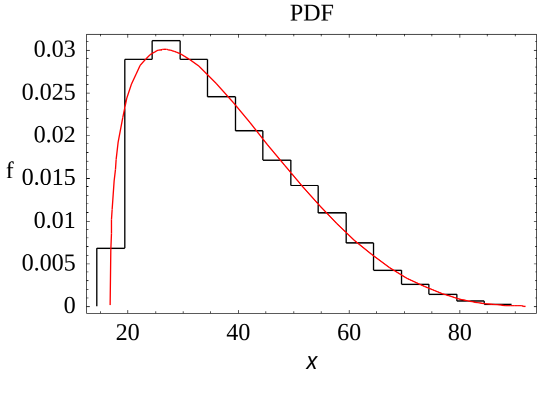

Fig. 6: The data 'histogram' and the smooth fitted Pearson pdf

≈ ***Example 3:*** Non-parametric kernel density estimation (NPKDE)

Here is some raw data measuring the diagonal length of 100 forged Swiss bank notes and 100 real Swiss bank notes (Simonoff, 1996).

```
data = Import["sd.dat"] // Flatten;
```

Nonparametric kernel density estimation involves two components: the choice of a kernel, and the selection of a bin-width.
Here we use a Gaussian kernel *f*:

$$f = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\,p}}; \qquad \texttt{domain[f] = \{x, -}\infty\texttt{, }\infty\texttt{\};}$$

Second, we select the **bin-width** *c*. Small values for *c* produce a rough estimate while large values produce a very smooth estimate. A number of methods exist to automate bin-width choice; the PDF package implements the Sheather-Jones (1991) approach. This can be used as a stand-alone bin-width selector, or, better still, as a starting point for experimentation. For the Swiss bank note data set, the Sheather-Jones optimal bin-width (using the Gaussian kernel *f*) is:

```
c = SheatherJones[data, f]
```

```
0.200059
```

We can now plot the smoothed non-parametric kernel density estimate using the `NPKDE[`*data*, *kernel*, *c*`]` function:

```
NPKDE[data, f, c];
```

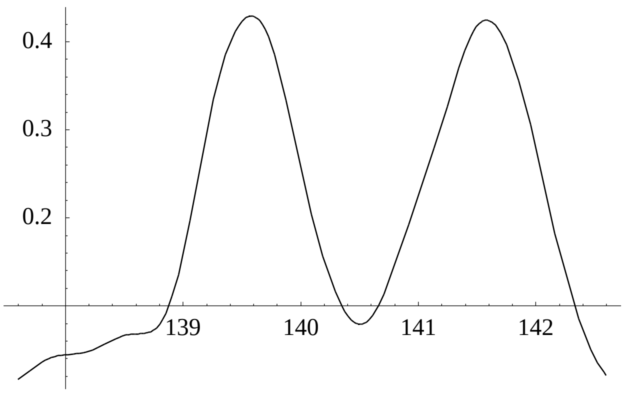

Fig. 7: The smoothed non-parametric kernel density estimate (Swiss bank notes)

≈ *Example 4:* Random number generation

Let X be *any* discrete random variable with probability mass function (pmf) $f(x)$. Then, the **PDF** package function `DiscreteRNG[n, f]` generates n pseudo-random copies of X. To illustrate, let us suppose $X \sim$ Poisson(6):

```
f = (e^-λ λ^x)/x! /. λ → 6;
domain[f] = {x, 0, ∞} && {Discrete};
```

As per usual, the term `domain[f]` must *always* be entered along with `f`, as it passes important information onto `DiscreteRNG`. Here are 30 copies of X:

```
DiscreteRNG[30, f]
```

```
{5, 7, 8, 2, 4, 7, 7, 6, 9, 11, 2, 5, 11,
 5, 4, 11, 5, 7, 3, 6, 5, 3, 9, 9, 9, 6, 3, 9, 7, 8}
```

Here are 50000 more copies of X:

```
data = DiscreteRNG[50000, f]; // Timing
```

```
{0.983333 Second, Null}
```

RNGDiscrete is not only completely general, but is also very efficient. We can contrast the empirical distribution of `data` with the true distributon of X:

```
FrequencyPlotDiscrete[data, f];
```

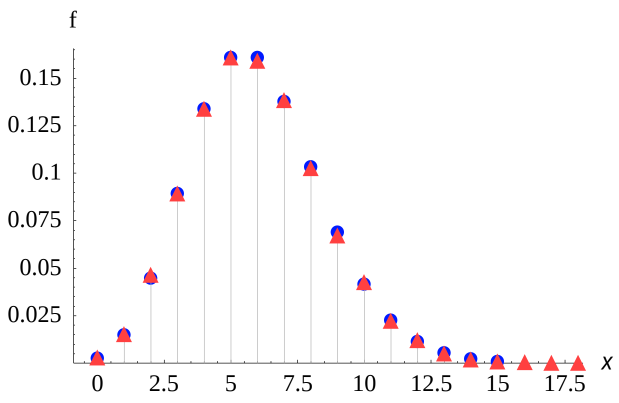

Fig. 8: The empirical pmf (triangles) and true pmf (round dots)

The triangular dots (red) denote the empirical pmf, whilst the round dots (blue) denote the true density $f(x)$. One obtains a superb fit because `DiscreteRNG` is an exact solution. This may make it difficult to distinguish the triangles from the round dots.

≈ *Example 5:* Moment conversion functions

The **PDF** package allows one to express any moment (raw h, central m, or cumulant k) in terms of any other moment (h, m, or k). For instance, to express the variance (the second central moment) $m_2 = E[(X - m)^2]$ in terms of raw moments, we enter:

```
CentralToRaw[2]
```

m_2 Æ $h_2 - h_1^2$

This is just the well-known result that $m_2 = E[X^2] - (E[X])^2$. As a further example, here is the sixth cumulant expressed in terms of raw moments:

```
CumulantToRaw[6]
```

k_6 Æ $-120\, h_1^6 + 360\, h_1^4\, h_2 - 270\, h_1^2\, h_2^2 + 30\, h_2^3 - 120\, h_1^3\, h_3 + 120\, h_1\, h_2\, h_3 - 10\, h_3^2 + 30\, h_1^2\, h_4 - 15\, h_2\, h_4 - 6\, h_1\, h_5 + h_6$

The moment converter functions are completely general, and extend in the natural manner to a multivariate framework. Here is the bivariate central moment $m_{2,3}$ expressed in terms of bivariate cumulants:

```
CentralToCumulant[{2, 3}]
```

$m_{2,3}$ Æ $6\, k_{1,1}\, k_{1,2} + k_{0,3}\, k_{2,0} + 3\, k_{0,2}\, k_{2,1} + k_{2,3}$

≈ *Example 6:* Unbiased estimation and moments of moments

The **PDF** package can find unbiased estimators of population moments. For instance, it handles h-statistics (unbiased estimators of population central moments), k-statistics (unbiased estimators of population cumulants), multivariate varieties of the same, polykays (unbiased estimators of products of cumulants) .... Consider the k-statistic k_r which is an unbiased estimator of the r^{th} cumulant k_r. Thus, $E[k_r] = k_r$, for $r = 1, 2, \ldots$. Here are the 2^{nd} are 3^{rd} k-statistics. As per convention, the solution is expressed in terms of power sums $s_r = \sum_{i=1}^{n} X_i^r$:

```
k2 = KStatistic[2]
k3 = KStatistic[3]
```

k_2 Æ $\dfrac{-s_1^2 + n\, s_2}{(-1 + n)\, n}$

k_3 Æ $\dfrac{2\, s_1^3 - 3\, n\, s_1\, s_2 + n^2\, s_3}{(-2 + n)\, (-1 + n)\, n}$

Moments of moments: with the **PDF** package, we can find any moment (raw, central, or cumulant) of the above expressions. For instance, k_3 is meant to have the property that

$E[k_3] = k_3$. We test this by taking the raw expectation of k_3, and express the answer in terms of cumulants:

```
RawExpectToCumulant[1, k3⟦2⟧]
```

k_3

In 1928, Fisher published the product cumulants of the k-statistics, which are now listed in reference bibles such as Stuart and Ord (1994). Here is the solution to $k_{2,2}(k_3,\ k_2)$ (Stuart and Ord - eqn 12.70):

```
CumulantExpectToCumulant[{2, 2}, {k3⟦2⟧, k2⟦2⟧}]
```

$$\frac{288\, n\, k_2^5}{(-2+n)\,(-1+n)^3} + \frac{288\,(-23+10\,n)\, k_2^2\, k_3^2}{(-2+n)\,(-1+n)^3} + \frac{360\,(-7+4\,n)\, k_2^3\, k_4}{(-2+n)\,(-1+n)^3} + \frac{36\,(160-155\,n+38\,n^2)\, k_3^2\, k_4}{(-2+n)\,(-1+n)^3\, n} + \frac{36\,(93-103\,n+29\,n^2)\, k_2\, k_4^2}{(-2+n)\,(-1+n)^3\, n} + \frac{24\,(202-246\,n+71\,n^2)\, k_2\, k_3\, k_5}{(-2+n)\,(-1+n)^3\, n} + \frac{2\,(113-154\,n+59\,n^2)\, k_5^2}{(-1+n)^3\, n^2} + \frac{6\,(-131+67\,n)\, k_2^2\, k_6}{(-2+n)\,(-1+n)^2\, n} + \frac{3\,(117-166\,n+61\,n^2)\, k_4\, k_6}{(-1+n)^3\, n^2} + \frac{6\,(-27+17\,n)\, k_3\, k_7}{(-1+n)^2\, n^2} + \frac{37\, k_2\, k_8}{(-1+n)\, n^2} + \frac{k_{10}}{n^3}$$

This is the correct solution; unfortunately, that given in Stuart and Ord (1994) and Fisher (1928) is actually incorrect. [The concerned reader will be pleased to know that we find full agreement with Stuart and Ord's other solutions, other than some small typographic errors.]

4 References and Acknowledgements

Fisher (1928), Moments and product moments of sampling distributions, *Proceedings of the London Mathematical Society*, series 2, volume 30, part 3, 199-238. Reprinted in Fisher, R.A.(1950), *Contributions to Mathematical Statistics*, Wiley, New York.

Sheather, S. J. and **Jones**, M. C. (1991), A reliable data-based bandwidth selection method for kernel density estimation, *Journal of the Royal Statistical Society, Series B*, (53), 683-690.

Simonoff, J.S. (1996), *Smoothing Methods in Statistics*, Springer: New York.

Stuart, A. and K. **Ord** (1994), *Kendall's Advanced Theory of Statistics*, volume 1, sixth edition, Edward Arnold (also John Wiley & Sons).

The PDF package was partly developed while Rose was visiting Wolfram Research, and while Smith was visiting the Sonderforschungsbereich 386, Institüt für Statistik, Ludwig-Maximilians-Universität, München. Rose gratefully acknowledges a Visiting Scholar Grant from Wolfram Research, while Smith acknowledges gratefully the support of the Alexander von Humboldt-Stiftung, and the Australian Research Council (through VISLAB and MEMLab). ■