WARREN D. MACEVOY

LUCK

PERSONAL NOTES

PUBLISHED BY PERSONAL NOTES

WWW.COLORADOMESA.EDU

First printing, May 2015

Contents

List of Figures

11 Given the log of the probabilities of a set of sample data, return a table used for quickly estimating luck. `problns` is a $1 \times n_{\text{nsamps}}$ row vector of logs of probabilities, and `eps` is an optional parameter giving the absolute error (in log space) for considering two probabilities to be equal. Returns `setup`, a $2 \times N$ matrix giving $-\log p(x)$ and estimates for $L(x)$ for each unique probability in the sample. This function is useful generally (not just for multinomial distributions). 24

12 Estimate luck given log of probabilities and setup. `problns` is a row vector of log-probabilities, `setup` is the setup from a (possibly different) sample. 24

13 Use the above functions for estimating luck numerically (`nluck`) and estimating the error in luck (numerical standard deviation). The last lines optionally compute the exact values of luck (`luck`) and standard deviations to compare with. One run produced `nluck=0.63025` and `luck=0.62875` with error bound estimates `nsd=0.004827`, `sd=0.0048314` and `z=0.3105`. 25

14 Histogram of z error values for 10,000 numerical approximations of luck. The maximum value of `sd` was `max(sd)=0.005`. 25

List of Tables

Introduction

The point of these notes is to introduce an idea of "luck" that connects mathematical probability with the everyday notion.

These might easily exist, perhaps as another name, in the literature; I just don't know what it is called.

As a motivating problem, imagine walking along the beach and asking a random person to toss a tennis ball so that it lands in the sand. The probability that it lands at some point would depend on the habits of the thrower and the details of the beach, but we can summarize this as some probability distribution, $p(x) = \rho(x)dx$, where $x \in \mathbb{R}^2$ is a suitable coordinate system for the beach in question. It would almost certainly not be a uniform distribution, and it would almost certainly not be particularly concentrated.

Traditional probability feels uncomfortable here. The chances of the ball landing at a given point is zero, and so miraculous. Yet anyone watching this process would only occasionaly be surprized by the outcome.

For a continuous probablity distribution such as this, the chance of the ball landing in some small area dx near x is $p(x) = \rho(x)dx$. But the ball lands at a point, so dx is zero, so the probablity $p(x) = \rho(x)dx$ is zero.

As common (and mundane, not miraculous) such situations are, the language of statistics seems to have difficulty with the notion. Nor is it limited to continuous cases, just when there are a lot of possible outcomes. Such examples lead to non-zero but very small probabilities.

The real motivation of this came from the space of passwords a person might choose from, which is an effectively infinite discrete space.

To distinguish from the more general notion of luck, note that that there is no extrinsic value on an outcome. To say something is "lucky" often means there is some value (different from the probability) associated with outcomes. However, outcomes that are the most valuable are often the least probable, and outcomes of equal probability ought to be equally lucky. In the most extreme case of all equally probable outcomes (uniform probability), every outcome should have a luck of $\frac{1}{2}$.

These observations lead to the following definition of luck:

> The luck $L(x)$ of an outcome x is the probability of getting any outcome y that is more probable than x, plus one-half the probability of getting any outcome y that is equally probable to x.

From the perspective of discussing luck, it is convenient to have few sets: $\Omega(x)$, the outcomes more likely than x, and $\omega(x)$, the outcomes equally likely to x.

Definition 1. Omega. $\Omega(x)$ is set of outcomes more likely than x:

$$\Omega(x) = \{y \mid p(y) > p(x)\} \,. \tag{1}$$

We define $|\Omega(x)|$ as the probability an outcome is in $\Omega(x)$, $|\Omega(x)| = P(y \in \Omega(x))$. In the discrete case, this is

$$|\Omega(x)| = \sum_{y \in \Omega(x)} p(y), \tag{2}$$

and, in the continuous case,

$$|\Omega(x)| = \int_{\Omega(x)} \rho(y)\, dy \,. \tag{3}$$

Definition 2. omega. $\omega(x)$ is the set of outcomes equally likley to x:

$$\omega(x) = \{y \mid p(y) = p(x)\} \,. \tag{4}$$

For the tyical case of many outcomes with different probabilities, $|\omega(x)|$ is small. For example, $|\omega(x)| = 0$ for any multivariate normal distribution.

Similar to $\Omega(x)$, we define $|\omega(x)|$ as the probability an outcome is in $\omega(x)$, $|\omega(x)| = P(y \in \omega(x))$.

In the discrete case, this is

$$|\omega(x)| = \sum_{y \in \omega(x)} p(y), \tag{5}$$

and, in the continuous case,

$$|\omega(x)| = \int_{\omega(x)} \rho(y)\, dy \,. \tag{6}$$

With these definitions in place, we define luck mathematically as follows:

Definition 3. Luck. The luck of an outcome is the probability getting any more likely outcome, plus half the probability of getting any equally likely outcome:

$$L(x) = |\Omega(x)| + \frac{1}{2}|\omega(x)| \,. \tag{7}$$

Properties of Luck.

- $0 \leq L(x) \leq 1$. This ranges from no luck to perfect luck.
- If $L(x)$ is close to 1, then $p(x)$ is comparatively small, and most outcomes would have a higher probability (you are lucky).
- If $L(x)$ is close to 0, then $p(x)$ is comparatively large, and most outcomes would have a lower probability (you are unlucky).
- $E(L) = \frac{1}{2}$. On average, luck is always 50:50.

We are interested in cases which have many possible outcomes with low but somewhat different probabilities (like the tennis ball on the beach). If the space is well divided (so $\max|\omega| = \max_x|\omega(x)|$ is small), then there are other interesting properties of luck:

- $E(f(L)) = \int_0^1 f(L)dL + \varepsilon$, where $|\varepsilon| \le \max|f''| \cdot \max|\omega|^2/12$.
- For $p \ge 1$, $E(L^p) = 1/(p+1) - \varepsilon$, $0 \le \varepsilon \le p \cdot (p-1)\max|\omega|^2/12$.
- For $0 \le a \le b \le 1$, $E(L \in [a,b]) = b - a + \varepsilon$, $|\varepsilon| \le \max|\omega|$.

In particular $|\omega(x)| = 0$ for the normal, exponential, beta, and gamma distributions. As a worst-case counterexample, the flattest distribution is the uniform distribution, for which $|\omega(x)| = 1$.

The proofs of these come from the midpoint integration rule and thinking about the general case for figure 1 below.

Example 1. Coins. Suppose we toss 8 fair coins. The probability of getting exactly x heads out of 8 tosses is given by the binomial distribution

$$p(x) = \frac{8!}{x!(8-x)!}\left(\frac{1}{2}\right)^8. \tag{8}$$

What is the luck associated with this distribution?

x	$p(x)$	$\Omega(x)$	$\lvert\Omega(x)\rvert$	$\omega(x)$	$\lvert\omega(x)\rvert$	$L(x)$
4	0.2734	{}	0.0000	{4}	0.2734	0.1367
3 or 5	0.2188	{4}	0.2734	{3,5}	0.4375	0.4922
2 or 6	0.1094	{3,4,5}	0.7109	{2,6}	0.2188	0.8203
1 or 7	0.0313	{2,3,4,5,6}	0.9297	{1,7}	0.0625	0.9609
0 or 8	0.0039	{1,2,3,4,5,6,7}	0.9922	{0,8}	0.0078	0.9961

Table 1: This is arranged in increasing luck (which is decreasing probability). Getting exactly $x = 4$ heads is unlucky, requiring only $L = 14\%$ luck, while getting $x = 0$ or $x = 8$ heads is almost 100% luck.

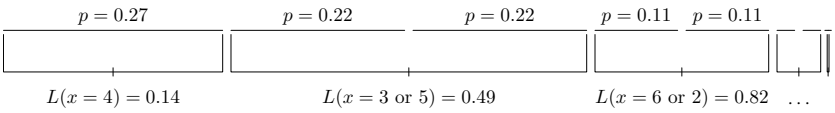

Figure 1: Luck to get x heads on 8 fair coins. After the probabilities are arranged in decreasing order on the unit interval, $L(x)$ is the center point of equally probable outcomes.

Luck on average is $\frac{1}{2}$:

$$E(L) = \sum_{x=0}^{8} p(x)\cdot L(x) = \frac{1}{2}. \tag{9}$$

For any distribution, $E(L) = \frac{1}{2}$.

The second moment should be close to $\frac{1}{3}$:

$$E(L^2) = \sum_{x=0}^{8} p(x)\cdot L(x)^2 = \frac{1}{3} - 0.0096 \tag{10}$$

For any distribution, $E(L^2) = \frac{1}{3} - \varepsilon$, with $0 \le \varepsilon \le \max|\omega|^2/6$.

For the coin distribution, the bound is $0 \le \varepsilon \le 0.032$, and the actual error $\varepsilon = 0.0096$.

The probability luck is in the middle half is about $\frac{1}{2}$:

$$\begin{aligned} E(L \in [\tfrac{1}{4},\tfrac{3}{4}]) &= \sum_{x=0}^{8} p(x)\cdot\left\{\begin{array}{ll} 1 & \text{if } L(x)\in[\frac{1}{4},\frac{3}{4}] \\ 0 & \text{otherwise}\end{array}\right\} \\ &= \frac{1}{2} - 0.0625. \end{aligned} \tag{11}$$

For any distribution, $E(L \in [a,b]) = b - a + \varepsilon$, with $|\varepsilon| \le \max|\omega|$.

For the coin distribution and $[a,b] = [\frac{1}{4},\frac{3}{4}]$, $E(L \in [a,b]) = \frac{1}{2} + \varepsilon$ with $|\varepsilon| \le 0.4375$ as the (poor) error bound, and the actual error of $\varepsilon = -0.0625$

Example 2. Normal. This is a special case of the multivariate normal we cover in the next section, but working out the details for the one-dimensional case can be illuminating. We define the one-dimensional normal distribution with mean μ and variance Σ as

$$P_{\text{normal}}(x;\mu,\Sigma) = \frac{e^{-\frac{(x-\mu)^2}{2\Sigma}}}{\sqrt{2\pi\Sigma}}\,, \tag{12}$$

where

$$\mu = E(x)\,, \tag{13}$$

and

$$\Sigma = E((x-\mu)^2)\,. \tag{14}$$

First note that $\Omega(x)$ is the open the interval between x and x reflected around μ:

$$\Omega(x) = (\min(x, 2\mu - x), \max(x, 2\mu - x))\,, \tag{15}$$

and $\omega(x)$ is the endpoints of that interval:

$$\omega(x) = \{x, 2\mu - x\}\,. \tag{16}$$

Since the 1-dimensional normal distribution is a continuous distribution and $\omega(x)$ is a finite set, $|\omega(x)| = 0$, i.e.,

$$|\omega(x)| = \int_{\omega(x)} P_{\text{normal}}(y;\mu,\Sigma)\,dy = 0\,. \tag{17}$$

This means all the luck properties are exact (the error terms are zero), and the $\frac{1}{2}|\omega(x)|$ contributes nothing to the luck of an outcome.

What remains is to calculate luck,

$$L(x) = \int_{\Omega(x)} P_{\text{normal}}(y;\mu,\Sigma)\,dy\,. \tag{18}$$

Changing variables to the normalized z-score: $z = \sqrt{\Sigma^{-1}}(x-\mu)$, this can be rewritten as

$$L(x) = \int_{-R}^{R} P_{\text{normal}}(y,0,1)\,dy\,, \tag{19}$$

where

$$R = |\sqrt{\Sigma^{-1}}(x-\mu)|\,. \tag{20}$$

Using $\text{erf}(x)$, defined as

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-y^2}\,dy\,, \tag{21}$$

$\text{erf}(x)$ is a normalized integral of the $P_{\text{normal}}(x;\mu = 0,\Sigma = 1/2)$ so that $\text{erf}(0) = 0$ and $\text{erf}(\pm\infty) = \pm 1$.

the luck of an outcome can be written as

$$L(x) = \text{erf}\left|\frac{x-\mu}{\sqrt{2\Sigma}}\right| . \tag{22}$$

In the next chapter, where we address the more general multivariate normal case, we obtain the approximation,

$$L(x) \approx \frac{1}{2}\left[1 + \text{erf}\left(\left|\frac{x-\mu}{\sqrt{\Sigma}}\right| - \sqrt{\frac{1}{2}}\right)\right] . \tag{23}$$

Figure 2 compares the exact and approximate result in the 1-d case.

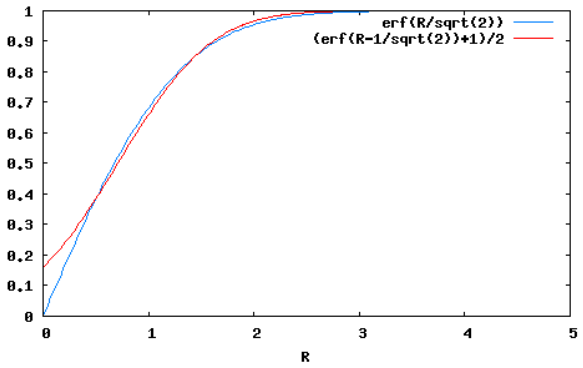

Figure 2: Exact (blue, equation 22) vs approximate (red, equation 23) luck for 1-d normal distribution.

Normal Distribution

Suppose we are in a probability space well approximated by the multivariate normal (Gaussian) distribution of a random variable $x \in \mathbb{R}^n$ with mean μ and non-singular covariance Σ:

The $n = 1$ example in the introduction is just where where μ and Σ are scalars.

$$P_{\text{normal}}(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}}{\sqrt{(2\pi)^n \det \Sigma}}\,, \tag{24}$$

where

$$\mu_i = E(x_i)\,, \tag{25}$$

and

$$\Sigma_{ij} = E((x_i - \mu_i)(x_j - \mu_j))\,. \tag{26}$$

How lucky is some outcome x? From the definition:

$$L(x) = |\Omega(x)| + \frac{1}{2}|\omega(x)|\,, \tag{27}$$

where

$$\Omega(x) = \{y \in \mathbb{R}^n | P_{\text{normal}}(y) > P_{\text{normal}}(x)\} \tag{28}$$

$$= \left\{y \in \mathbb{R}^n \middle| |\sqrt{\Sigma^{-1}}(y-\mu)| < |\sqrt{\Sigma^{-1}}(x-\mu)|\right\} \tag{29}$$

and

$$\omega(x) = \left\{y \in \mathbb{R}^n \middle| |\sqrt{\Sigma^{-1}}(y-\mu)| = |\sqrt{\Sigma^{-1}}(x-\mu)|\right\}\,. \tag{30}$$

Because $\omega(x)$ has no volume in $\mathbb{R}^n$,

$$|\omega(x)| = \int_{\omega(x)} P_{\text{normal}}(y; \mu, \Sigma)\, dy = 0\,. \tag{31}$$

So

$$L(x) = |\Omega(x)| \tag{32}$$

$$= \int_{\Omega(x)} P_{\text{normal}}(y; \mu, \Sigma)\, dy\,. \tag{33}$$

By changing variables to $z = \sqrt{\Sigma^{-1}}(x - \mu)$,

Σ is symmetric and positive definite, and so is its inverse. The square-root can be computed as a Cholesky decomposition. In Scilab, z=**chol**(Sigma)'(x−mu)

$$L(x) = \int_{|z|<R} P_{\text{normal}}(z; 0, I)\, dz\,, \tag{34}$$

where $R = |\sqrt{\Sigma^{-1}}(x - \mu)|$.

R=**norm**(z)

This can be evaluated in spherical coordinates:

$$L(x) = \frac{1}{\sqrt{(2\pi)^n}} \int_0^R \frac{n\pi^{n/2}}{\Gamma(\frac{n}{2}+1)} r^{n-1} e^{-\frac{1}{2}r^2}\, dr\,, \tag{35}$$

$$= \frac{\gamma(n/2, R^2/2)}{\Gamma(n/2)}\,. \tag{36}$$

The last form uses the lower incomplete gamma function and gamma function, defined to be

L=**cdfgam**("PQ",R^2/2,n/2,1)

$$\gamma(s, x) = \int_0^x t^{s-1} e^{-t}\, dt \tag{37}$$

$$\Gamma(s) = \gamma(s, \infty)\,. \tag{38}$$

For any value of n, but particularly for large values, we find the following approximation to be very good:

This comes from a Taylor expansion of the log of the integrand in (35), which is specifically invalid for $n = 1$, hence the difference between the general case and the $n = 1$ case (22).

Lapprox=0.5*(1+**erf**(R−**sqrt**(n−1/2)))

$$L(x) \approx \frac{1}{2}\left[1 + \text{erf}(|\sqrt{\Sigma^{-1}}(x - \mu)| - \sqrt{n - 1/2})\right]\,. \tag{39}$$

Not only is this result pretty, it is very useful. Suppose we have distribution parameters μ and Σ, and would like to know if they fit actual observations. A traditional approach requires a large sample to estimate μ and Σ, but we don't need this, nor do we need to assume that the distribution is normal. We just need to ask if the observations are surprising (lucky or unlucky). In large dimensions, numerical experiments suggest one sample is in most cases sufficient to establish practical certainty (probability of error less than 10^{-15}).

The approximate result (39) leads to a rule for combining (normal) luck: Suppose there are two independent normal distributions parameterized by $(\mu^{(x)}, \Sigma^{(x)})$ of dimension n_x, and $\mu^{(y)}$, $\Sigma^{(y)}$ of dimension n_y. What is the luck of a single combined observation (x, y)?

$$L(x, y) \approx \frac{1}{2}\left[1 + \text{erf}(R_{xy}(x, y) - \bar{R}_{xy})\right]\,, \tag{40}$$

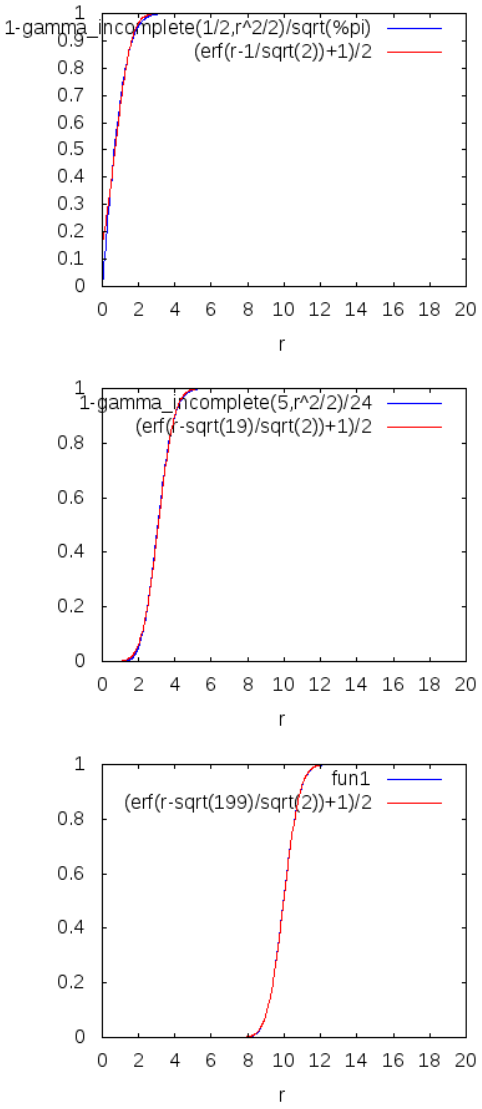

Figure 3: Exact (blue) vs approximate (red) luck for normal distribution for $n = 1$, 10, and 100.

$L^{(x)}(x)$	$L^{(y)}(x)$	$L^{(x)}(y)$	$L^{(y)}(y)$
0.501 417 202 0	1.000 000 000 0	1.000 000 000 0	0.838 180 264 1
0.731 421 266 5	1.000 000 000 0	1.000 000 000 0	0.239 258 143 2
0.982 563 033 9	1.000 000 000 0	1.000 000 000 0	0.271 695 512 7
0.033 455 080 7	1.000 000 000 0	1.000 000 000 0	0.421 320 625 9
0.689 429 934 0	1.000 000 000 0	1.000 000 000 0	0.074 461 655 7
0.736 397 593 7	1.000 000 000 0	1.000 000 000 0	0.294 050 728 4
0.304 521 296 7	1.000 000 000 0	1.000 000 000 0	0.707 849 014 7
0.231 111 574 4	1.000 000 000 0	1.000 000 000 0	0.290 313 093 2
0.585 247 719 9	1.000 000 000 0	1.000 000 000 0	0.636 902 202 8
0.214 552 926 1	1.000 000 000 0	1.000 000 000 0	0.268 989 787 4

Table 2: Luck from two randomly generated distributions $\mu^{(x)}$ and $\mu^{(y)}$ uniformly chosen in $[0,1]^{100}$, and $\Sigma^{(x)}, \Sigma^{(y)}$ are transposed squares of random 100×100 matrices. In each row, x is a sample from the $\mu^{(x)}, \Sigma^{(x)}$, normal distribution, and y is from the $\mu^{(y)}, \Sigma^{(y)}$ distribution. The actual values of x and y are not given, since they are very large (100 numbers each) and uninteresting.

where

$$R_{xy}(x,y) = \sqrt{R_x(x)^2 + R_y(y)^2}\,, \tag{41}$$

$$\bar{R}_{xy} = \sqrt{\bar{R}_x^2 + \bar{R}_y^2 + \frac{1}{2}}\,, \tag{42}$$

$$R_x(x) = \sqrt{\left(\Sigma^{(x)}\right)^{-1}}\left(x - \mu^{(x)}\right)\,, \tag{43}$$

$$R_y(x) = \sqrt{\left(\Sigma^{(y)}\right)^{-1}}\left(y - \mu^{(y)}\right)\,, \tag{44}$$

$$\bar{R}_x = \sqrt{n_x - \frac{1}{2}}\,, \tag{45}$$

$$\bar{R}_y = \sqrt{n_y - \frac{1}{2}}\,. \tag{46}$$

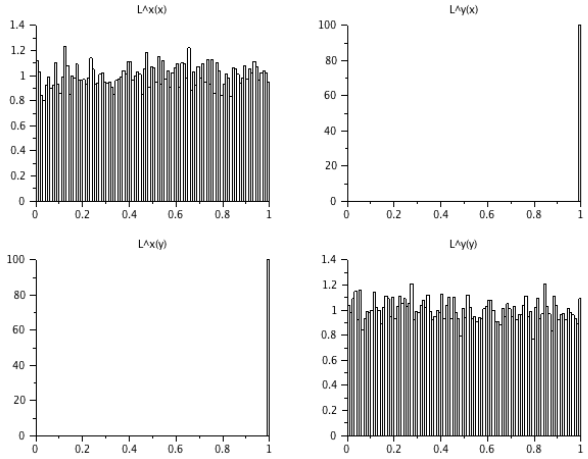

Figure 4: Histograms of 10,000 luck values in the same distributions as table 2. Upshot: y values in the x distribution are viewed as extremely lucky and vice-versa, while they are have uniform luck in their own respective distributions.

The listings in figures 5-7 give Scilab functions for basic luck calculations for multivariate normal distributions.

```
function lnp=mnprobln(x,mu,Sigma)
  [dim,nsamps]=size(x);
  sigma=chol(Sigma)';
  z=sigma\(x-mu*ones(1,nsamps));
  R2=sum(z.^2,'r');
  lnp=-R2/2-(dim/2*log(2*%pi)+...
       sum(log(diag(sigma))));
endfunction
```

Figure 5: Scilab listing to compute the natural log of the probability of multivariate normal outcomes. x is a $n_{\text{dim}} \times n_{\text{samps}}$ of outcomes, μ is a $n_{\text{dim}} \times 1$ column vector of the means, and Σ is a $n_{\text{dim}} \times n_{\text{dim}}$ covariance matrix. The result `lnp` is a $1 \times n_{\text{samps}}$ row vector.

```
function L=mnluck(x,mu,Sigma)
  [dim,nsamps]=size(x);
  one=ones(1,nsamps);
  sigma=chol(Sigma)';
  z=sigma\(x-mu*one);
  R=sqrt(sum(z.^2,'r'));
  L=cdfgam("PQ",R.^2/2,...
       dim/2*one,one);
endfunction
```

Figure 6: Scilab listing to efficiently compute the luck of multivariate normal outcomes. x is a $n_{\text{dim}} \times n_{\text{samps}}$ of outcomes, μ is a $n_{\text{dim}} \times 1$ column vector of the means, and Σ is a $n_{\text{dim}} \times n_{\text{dim}}$ covariance matrix. The result L is a $1 \times n_{\text{samps}}$ row vector of the luck associated with each outcome.

```
function L=mnapproxluck(x,mu,Sigma)
  [dim,nsamps]=size(x);
  sigma=chol(Sigma)';
  z=sigma\(x-mu*ones(1,nsamps));
  R=sqrt(sum(z.^2,'r'));
  L=0.5*(1+erf(R-sqrt(dim-0.5)));
endfunction
```

Figure 7: Like `mnluck` in figure 6, but approximates luck via (39).

Computation

Unfortunately, the explicit computations for luck may be impractical. However, if n independent samples are taken $S = (x_1, \ldots, x_n)$, then it can be estimated as:

$$\begin{aligned}\ell(x) =& \frac{1}{n}\{\text{\# of outcomes in S more probable than } x\} \\ &+ \frac{1}{2n}\{\text{\# of outcomes in S equally probable to } x\}\end{aligned}$$

It is a reasonably straightforward calculation to show that

$$E(\ell(x)) = L(x)\,, \tag{47}$$

and

$$E((\ell(x) - L(x))^2) \leq \frac{1}{n}L(x)\cdot(1 - L(x)) \leq \frac{1}{4n}\,. \tag{48}$$

Example 3. Multinomial. We are aware of no computationally efficient way to exactly compute the luck for a multinomial distribution other than the explicit sum. Suppose there is a questionnaire with 4 possible answers with category probabilities 0.1, 0.2, 0.3, and 0.4. How lucky would it be to get 13, 15, 27 and 45 responses of each respective answer in 100 samples?

The log-probability in this case is the multinomial probability (see figure 8):

$$y = \text{mulprobln(x,p)} = \log\left[\left(\sum_{i=1}^{n_p} x_i\right)!\prod_{i=1}^{n_p}\frac{p_i^{x_i}}{x_i!}\right]\,. \tag{49}$$

Numerical estimates of luck requires samples from the given probability distribution. Figure 9 uses a built-in function in scilab to generate such a sample set.

For smaller spaces, luck for a multinomial distribution can be computed explicitly. Figure 10 computes this as an inefficient testing reference.

```
function lnp=mulprobln(x,p)
  [nprobs,nsamps]=size(x);
  pln=log(p);
  y=zeros(1,nsamps);
  for i=1:nsamps
    xi=x(:,i);
    ni=sum(xi);
    lnp(i)=gammaln(ni+1)+...
      sum(xi.*pln-gammaln(xi+1));
  end
endfunction
```

Figure 8: Scilab function to compute the natural log of the probability of multinomial outcomes. p is a $n_{\text{probs}} \times 1$ column vector of category probabilities, x is a $n_{\text{probs}} \times n_{\text{samps}}$ matrix of outcomes, and the result `lnp` is a $1 \times n_{\text{samps}}$ row vector.

```
function x=mulsamp(nsamps,ntrials,p)
  x=grand(nsamps,"mul",ntrials,p(1:(length(p)-1)));
endfunction
```

Figure 9: Scilab listing to get a sample of outcomes from a multinomial distribution. n_{samps} is the number of desired samples, n_{trials} is the number of trials in each sample, and p is a $n_{\text{probs}} \times 1$ column vector of category probabilities. The result x is a $n_{\text{probs}} \times n_{\text{samps}}$ matrix of sample outcomes, where $\sum_{j=1}^{n_{\text{probs}}} x(j,i) = n_{\text{trials}}$.

```
// Used by mulluck below to
// recursively compute luck
function el=mulluckrec(nb,mb,...
                lpx,k,p,y,eps)
  [n,m]=size(lpx);
  el=zeros(n,m);
  for yk=0:nb-mb
     y(k)=yk;
     if k < length(p)-1 then
       el=el+mulluckrec(nb,mb+yk,...
         lpx,k+1,p,y,eps);
     else
       y(length(p))=nb-mb-yk;
       lpy=mulprobln(y,p);
       c=0.5*bool2s(lpy > lpx-eps)+...
            0.5*bool2s(lpy > lpx+eps);
       el = el + c .* exp(lpy);
     end
  end
endfunction

function L=mulluck(x,p,eps)
  if ~exists("eps","local") then
    eps=sqrt(%eps);
  end
  [nprobs,nsamps]=size(x);
  ntrials=sum(x,'r');
  no=min(ntrials);
  n1=max(ntrials);
  assert_checkequal(no,n1);
  L=mulluckrec(no,0,mulprobln(x,p),...
      1,p,zeros(nprobs,1),eps);
endfunction
```

Figure 10: Recursively compute luck of multinomial exactly using exhaustive sum. x is a $n_{\text{probs}} \times n_{\text{samps}}$ matrix of outcomes, and p is a $n_{\text{probs}} \times 1$ column vector of category probabilities. The result is a $1 \times n_{\text{samps}}$ of luck values.

```
function setup=numlucksetup(problns,eps)
  if ~exists("eps","local") then
    eps=sqrt(%eps);
  end

  n=length(problns);
  problns=gsort(problns);

  for pass=1:2
    if pass == 2 then
      setup=zeros(2,count);
    end
    i=1;
    count=0;
    while i<=n
      j=i;
      while j <= n-1 & ...
          problns(i)-problns(j+1) < eps
        j=j+1;
      end
      count=count+1;
      if pass == 2 then
        setup(1,count)= ...
          -0.5*(problns(i)+problns(j));
        setup(2,count)= ...
          (i-1)/n+0.5*(j-i+1)/n;
      end
      i=j+1;
    end
  end
endfunction
```

Figure 11: Given the log of the probabilities of a set of sample data, return a table used for quickly estimating luck. `problns` is a $1 \times n_{\text{nsamps}}$ row vector of logs of probabilities, and `eps` is an optional parameter giving the absolute error (in log space) for considering two probabilities to be equal. Returns `setup`, a $2 \times N$ matrix giving $-\log p(x)$ and estimates for $L(x)$ for each unique probability in the sample. This function is useful generally (not just for multinomial distributions).

```
function L=numluck(problns,setup)
  L=max(0,min(1,interpln(setup,-problns)));
endfunction
```

Figure 12: Estimate luck given log of probabilities and setup. `problns` is a row vector of log-probabilities, `setup` is the setup from a (possibly different) sample.

```
// get sample set
nsamps=10000;
ntrials=100;
p=[0.1;0.2;0.3;0.4];
x=mulsamp(nsamps,ntrials,p);
problns=mulprobln(x,p);

// setup for luck estimates
setup=numlucksetup(problns);

// estimate luck numerically
x0=[13; 15; 27; 45];
problns0=mulprobln(x0,p);
nluck=numluck(problns0,setup);
nsd=sqrt(nluck .* (1-nluck) ./ nsamps);

// optional exact luck
luck=mulluck(x0,p);
sd=sqrt(luck .* (1-luck) ./ nsamps);

// z is approximately normally disributed
z=(nluck-luck) ./ sd;
```

Figure 13: Use the above functions for estimating luck numerically (`nluck`) and estimating the error in luck (numerical standard deviation). The last lines optionally compute the exact values of luck (`luck`) and standard deviations to compare with. One run produced `nluck=0.63025` and `luck=0.62875` with error bound estimates `nsd=0.004827`, `sd=0.0048314` and `z=0.3105`.

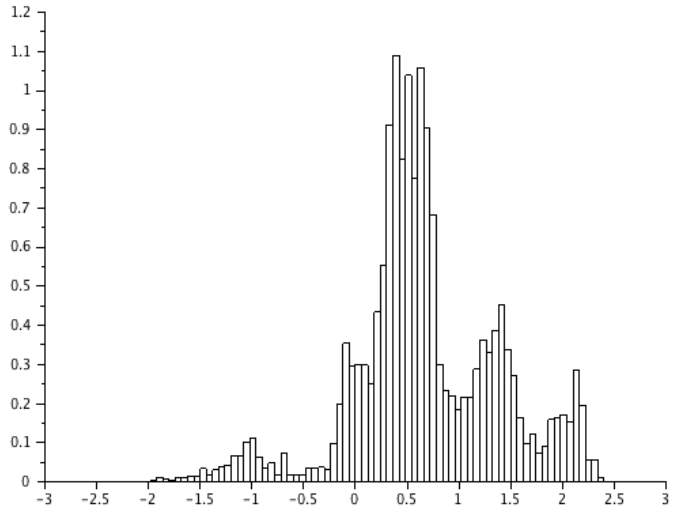

Figure 14: Histogram of z error values for 10,000 numerical approximations of luck. The maximum value of `sd` was `max(sd)=0.005`.