Truncated Normal Collocation: (Chasing the One-Armed Man)

John Burkardt
Department of Scientific Computing
Florida State University

....

3:30-4:45pm, 14 February 2014,
Graduate Student Seminar ISC5934

....

http://people.sc.fsu.edu/∼jburkardt/presentations/
truncated_normal_2014_fsu.pdf

INTRO:

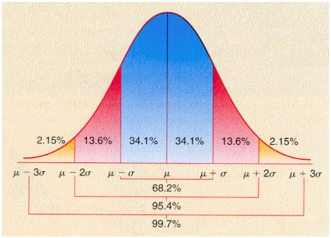
In a prehistoric TV series called "The Fugitive", Dr Richard Kimble, falsely accused of murdering his wife, searched for the one-armed man who was the real killer.

There are no murders scheduled for today, but I will be lopping off the right arm, the left arm, or both arms, of the standard normal distribution!

Truncated Normal Collocation

- **The Strange Case of the Abnormal Normal**
- Can You Describe the Suspect?
- I Have a Cunning Plan
- A Matter of Moments
- The Summing Up

ABNORMAL: The Normal Probability Distribution



I was recently invited to Ajou University, Korea, at the invitation of Professor Hyung-Chun Lee. One morning during my visit, he asked me if I could set up a collocation procedure for the truncated normal distribution, and I said, *"Sure, no problem!"*

The truncated normal distribution is a simple modification to our familiar friend, the normal distribution.

The normal distribution allows a natural description of how some measurable quantities (height, income, number of sick days) have a dominant average value μ, and an associated tendency to vary, called σ^2.

ABNORMAL: The Truncated Normal

Mathematical models are idealizations, and have their limitations. If we think the normal distribution is a good model of height distribution, then strictly speaking, we are admitting the possibility (small, but not zero!) of people who are as tall as 60 or 1000 feet - or negative 200 feet, for that matter.

This discrepancy could be a problem if we are doing a simulation, for instance. Then we treat the mathematical distribution as physcial reality, we sample it, and we "believe" whatever comes out of the process. If we create a 1-inch person, then we are now stuck dealing with a physically meaningless but computationally real object.

ABNORMAL: The Truncated Normal

Sometimes these 1 inch people can actually cause the computation to crash, or to produce meaningless results.

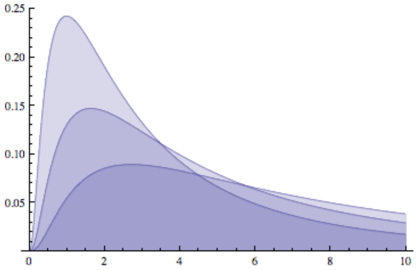
In our research group, a commonly studied problem involves the simulation of the permeability function $a(\omega, x)$ related to groundwater flow, arising in the equation:

$$\nabla \cdot (a(\omega, x) \nabla u(x)) = f(x)$$

Mathematically, a solution to this problem can be guaranteed to exist as long as the permeability function is everywhere positive, bounded away from 0 and from infinity:

$$0 < a_{min} \leq a(\omega, x) \leq a_{max} < \infty$$

ABNORMAL: The Log-Normal Probability Distribution



Sometimes, instead of using the normal probability density function to describe the randomness in $a(\omega, x)$, researchers use the log-normal PDF, because if we assume that

$$log(a()) = \alpha \sim N(\mu, \sigma)$$

then we are guaranteed that

$$0 < e^{\alpha} = a()$$

ABNORMAL: The Log-Normal Probability Distribution

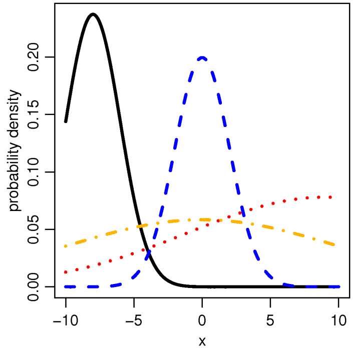
If we use the log-normal PDF in this way, our simulation will never select a negative value for $a()$, and $a()$ is described by a mathematically tractable and plausible formula.

But while $a()$ can't be zero, it can get arbitrarily small or large, meaning the mathematical requirement was not really guaranteed, and numerically we may have issues with stability or ill-conditioning.

The distribution is also clearly a different shape from the normal distribution; it's not symmetric, and has different variance properties,

Rather than go to a new distribution, researchers have also considered keeping the normal distribution, but altering it in some simple way to avoid the problem areas.

ABNORMAL: The Truncated Normal Distribution



A natural modification to the normal distribution restricts the PDF by defining an upper maximum, or a lower minimum or both.

ABNORMAL: The Truncated Normal Distribution

This is a little like shooting at a (finite) paper target. If we assume our aim has a normal distribution of error, then our shots will tend to cluster near the center, but trail off forever. However, we only keep statistics on the shots that actually hit the paper target.

We can do this, but since mathematics is our guide, we should justify our computations by clearly defining this new distribution and working out its properties.

ABNORMAL: Parameters

For flexibility, a PDF is described by parameters that allow us to specify that some quantities are choices we will make later.

Since a truncated normal PDF starts life as a normal PDF, two parameters we must must supply are the values of $\overline{\mu}$ and $\overline{\sigma}$ that specify that parent normal.

Our remaining two choices define the truncation interval $[a, b]$, and we should allow for all four possibilities:

- $(-\infty, +\infty)$, standard or non-truncated normal;
- $[a, +\infty)$, lower truncated normal;
- $(-\infty, b]$, upper truncated normal;
- $[a, b]$, doubly truncated normal.

So the parameters for a truncated normal are $\overline{\mu}, \overline{\sigma}, a, b$ and we may write it as $\text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x)$

Note that $\overline{\mu}$ is not the mean μ of the truncated normal (we'll have to figure that out), and $\overline{\sigma}^2$ is not its variance σ^2 (another thing we'll have to work out)!

Truncated Normal Collocation

- The Strange Case of the Abnormal Normal
- **Can You Describe the Suspect?**
- I Have a Cunning Plan
- A Matter of Moments
- The Summing Up

DESCRIBE: PDF/CDF/invCDF/SAMPLE/MEAN/VAR

Suppose we are given the vital statistics of a truncated normal distribution, namely, the values of $\overline{\mu}, \overline{\sigma}, a, b$.

There are six standard chores we should be able to do:

1. evaluate $\text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x)$;
2. evaluate $\text{cdf}(\overline{\mu}, \overline{\sigma}, a, b; x) = \int_a^x \text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; \xi) d\xi$;
3. solve $C = \text{cdf}(\overline{\mu}, \overline{\sigma}, a, b; x)$ for the value of x;
4. uniformly sample $\text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x)$.
5. evaluate $\mu = \int_a^b x \, \text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x) dx$;
6. evaluate $\sigma^2 = \int_a^b (x - \mu)^2 \text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x) dx$;

DESCRIBE: PDF

Task 1: evaluate $\psi(x) = \text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x)$;

Denote by $\phi(\xi)$ and $\Phi(\xi)$ the PDF and CDF for the standard normal distribution with mean 0 and variance 1.

To adjust for the effects of the nonstandard mean and variance, define:

$$\xi(x) = \frac{x - \overline{\mu}}{\overline{\sigma}}$$

Then we can normalize the PDF over the nontruncated range:

$$\psi(x) = \begin{cases} 0 & \text{if } x < a \\ \frac{\phi(\xi(x))}{\Phi(\xi(b)) - \Phi(\xi(a))} & \text{if } a \le x \le b \\ 0 & \text{if } b < x \end{cases}$$

The quantity

$$S = \Phi(\xi(b)) - \Phi(\xi(a))$$

is a scale factor which we will need in order to normalize our integrals involving the truncated normal PDF.

DESCRIBE: CDF

Task 2: evaluate $\Psi(x) = \text{cdf}(\overline{\mu}, \overline{\sigma}, a, b; x) = \int_a^x \text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; \xi) d\xi$.

We know that the CDF evaluated at a point x is simply the integral of the PDF up to that value. So we can simply integrate the formula from the previous task. We'll assume that $a \leq x \leq b$:

$$\begin{aligned}
\Psi(x) &= \int_{-\infty}^{x} \psi(x) dx = \int_{a}^{x} \psi(x) dx \\
&= \int_{a}^{x} \frac{\phi(\xi(x))}{\Phi(\xi(b)) - \Phi(\xi(a))} dx \\
&= \frac{\int_a^x \phi(\xi(x)) dx}{\Phi(\xi(b)) - \Phi(\xi(a))} \\
&= \frac{\Phi(\xi(x)) - \Phi(\xi(a))}{\Phi(\xi(b)) - \Phi(\xi(a))}
\end{aligned}$$

DESCRIBE: CDF

The CDF has to be 0 at a and 1 at b, and in between it's simply integrating the scaled PDF. So the formula has to be:

$$\Psi(x) = \begin{cases} 0 & \text{if } x < a \\ \frac{\Phi(\xi(x)) - \Phi(\xi(a))}{S} & \text{if } a \le x \le b \\ 1 & \text{if } b < x \end{cases}$$

DESCRIBE: invCDF

Task 3: solve $C = \text{cdf}(\overline{\mu}, \overline{\sigma}, a, b; x)$ for the value of x;

We can almost solve the previous CDF equation:

$$\Phi(\xi(x)) = S * C + \Phi(\xi(a))$$

Presumably, we can invert the CDF of the normal distribution:

$$\xi(x) = \Phi^{-1}(S * C + \Phi(\xi(a)))$$

and so the corresponding value of x is simply:

$$x = \overline{\mu} + \overline{\sigma}\,\xi$$

DESCRIBE: SAMPLE

Task 4: uniformly sample pdf$(\overline{\mu}, \overline{\sigma}, a, b; x)$;

Luckily, solving Task 3 makes this task trivial. To sample from the distribution, simply generate a uniform random value $C \in [0, 1]$. Regard C as the value of the CDF at a point x, and compute x.

Values chosen in this way are uniformly distributed with respect to the truncated normal distribution.

By the way, there is an alternative sampling method that is correct, and way simpler.

This simple method samples a value from the standard normal distribution, pdf$(\overline{\mu}, \overline{\sigma}, -\infty, +\infty; x)$ but if the value is less than a or greater than b, we reject it, and try again.

DESCRIBE: SAMPLE (Bad Approach)

The nice thing about the simple "rejection" method is that it is trivial to program. If you can generate random normal numbers, you can generate random truncated normal numbers by throwing away the ones you can't use.

The drawback is that the amount of work needed to compute a sample grows enormously as the truncation interval moves around.

We might imagine our typical truncation interval is something like [-5,+5] so we only throw away data that is 5 standard deviations off.

But we should be able to use the same program to sample in the interval [+5,+6], in which case we need to throw away all the data that is **less than** 5 standard deviations from the mean, and more. That program will take forever to run.

So we really would prefer a sampling method that takes about the same time to produce 1,000 samples, no matter what truncation interval we specify. And we can do this.

DESCRIBE: MEAN

Task 5: evaluate $\mu = \int_a^b x \, \text{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x) dx$;

A formula is available for this task:

$$\mu = \overline{\mu} + \frac{\phi(\alpha) - \phi(\beta)}{S} \overline{\sigma}$$

where

$$\alpha = \frac{a - \overline{\mu}}{\overline{\sigma}}; \quad \beta = \frac{b - \overline{\mu}}{\overline{\sigma}};$$

For the lower truncated normal, we have $b = \infty$ so $\phi(\beta) = 0$; we can handle the upper truncated normal similarly, and we see that the formula will also be correct for the normal distribution as well, simply returning $\mu = \overline{\mu}$.

DESCRIBE: VARIANCE

Task 6: evaluate $\sigma^2 = \int_a^b (x - \mu)^2 \mathrm{pdf}(\overline{\mu}, \overline{\sigma}, a, b; x) dx$;

A formula is also available for this task:

$$\sigma^2 = \overline{\sigma}^2 (1 + \frac{\alpha\phi(\alpha) - \beta\phi(\beta)}{S} - (\frac{\phi(\alpha) - \phi(\beta)}{S})^2)$$

Again, the formula is written for the doubly-truncated case, but can easily be used for the lower, upper, and non-truncated distributions as well.

DESCRIBE: Checking the Facts

I set up a library called **truncated_normal** which included code for the six tasks, for all four truncation possibilities,

Now I figured I needed some confidence in my formulas before moving on, so I constructed a set of tests.

One simple test is to start with a value of X, compute its CDF, then compute invCDF and see if we get back to X.

The second test was to do a simulation. That is, use the SAMPLE function to compute, say, 10,000 sample values of a distribution, compute the sample mean and variance, and compare them to the MEAN and VARIANCE functions.

After banging on the code, the tests behaved, and I felt much better.

http://people.sc.fsu.edu/∼jburkardt/m_src/truncated_normal/truncated_normal.html

DESCRIBE: You're Not Done!

I kept busy all morning long working out these details about the truncated normal distribution and trying to program, document, and test them.

That afternoon, Professor Lee came back into the office and asked if I had been able to complete the collocation task.

"Well...", I said hesitantly, *"I can PDF, CDF, invCDF, SAMPLE, MEAN and VARIANCE."*

"What about the collocation?" he asked.

"Actually," I said, *"I might need one more day..."*

As it turned out, I ended up working on this problem for another month.

Truncated Normal Collocation

- The Strange Case of the Abnormal Normal
- Can You Describe the Suspect?
- **I Have a Cunning Plan**
- A Matter of Moments
- The Summing Up

PLAN: What Do We Need?

We weren't pursuing the truncated normal distribution for its own sake - what we were really after was the ability to do collocation.

We wanted to estimate the expected value of "quantities of interest" associated with a system of partial differential equations that included stochastic input terms.

The stochastic input terms were going to be modeled by truncated normal distributions, so that they behaved like normal variables, but over a truncated range.

In order to estimate the quantities of interest, a collocation procedure selects many test values of the input terms, weighted by their probabilities, and computes an average that is really a multidimensional integral.

PLAN: We Need a Quadrature Rule

If every input term is controlled by a truncated normal distribution, then the crucial tool we need is a sequence of quadrature rules, of increasing accuracy, for that distribution.

A quadrature rule for the truncated normal distribution is a set of n points x_i and weights w_i for which we make the estimate

$$\frac{1}{S\sqrt{2\pi\overline{\sigma}^2}} \int_a^b f(x) e^{-\frac{(x-\overline{\mu})^2}{2\overline{\sigma}^2}}\, dx \approx \sum_{i=1}^{n} w_i \cdot f(x_i)$$

We are usually looking for a quadrature rule of Gaussian type, so that the n-point rule will integrate precisely any function $f(x)$ which is a polynomial of degree $2n-1$ or less.

Analytic formulas are known for special weight functions; otherwise, a famous paper by Golub and Welsch shows how to construct a matrix whose eigendecomposition will produce the desired rule.

PLAN: Orthogonal Polynomial Family

The most common algorithm described by Golub and Welsch assumes that the user knows a family of polynomials $\phi_i(x), i = 0, ...$ which are orthogonal with respect to the PDF.

To explain what is going on, let us suppose that we use our PDF to define a new inner product of any two functions $f()$ and $g()$ by:

$$< f, g > \equiv \int_{-\infty}^{+\infty} f(x)g(x) \text{ pdf}(\overline{\mu}, \overline{\sigma}, a, b; x) dx$$

We can define a corresponding function norm:

$$||f|| = \sqrt{< f, f >}$$

in which case we will want to restrict our attention to the space of all functions whose norm is finite.

PLAN: Orthogonal Polynomial Family

If we can get a basis for this space, we know a lot about how it works. It is natural to analyze functions in terms of polynomials. A family of orthogonal polynomials $\phi_i(x)$ with respect to a given PDF is exactly an orthogonal basis for the given space, so that:

$$\int_{-\infty}^{+\infty} \phi_i(x)\phi_j(x) \text{ pdf}(\overline{\mu}, \overline{\sigma}, a, b; x)dx = \delta_{i,j}$$

(This formula assumes that I have taken one more step, and divided each basis function by its norm.)

So these basis polynomials give a way of understanding all the elements of the space, and are similar to thinking of the column vectors of the identity matrix as a basis for all vectors in $\mathbb{R}^n$.

PLAN: Orthogonal Polynomial Family

For the normal distribution, the orthogonal family is the Hermite polynomials:

$$\begin{aligned} H_0(x) =& 1 \\ H_1(x) =& x \\ H_2(x) =& x^2 - 1 \\ H_3(x) =& x^3 - 3x \end{aligned}$$

Orthogonal polynomial families must satisy a three term recurrence:

$$\phi_{i+1}(x) = \alpha_i \, x \, \phi_i(x) + \beta_i \, \phi_{i-1}(x)$$

and for the Hermite polynomials, $\alpha_i = 1$ and $\beta_i = -i$, so that

$$\begin{aligned} H_2(x) =& x \, H_1(x) - 1 \, H_0(x) = x^2 - 1 \\ H_3(x) =& x \, H_2(x) - 2 \, H_1(x) = x^3 - 3x \\ H_4(x) =& x \, H_3(x) - 3 \, H_2(x) = \text{(can you fill this in?)} \end{aligned}$$

...and so on

PLAN: Quadrature by Eigenvalues

In cases where we can determine the recurrence coefficients, the Golub-Welsch procedure forms what is known as the Jacobi matrix:

$$J = \begin{pmatrix} \alpha_0 & \sqrt{\beta_1} & 0 & \dots & 0 \\ \sqrt{\beta_1} & \alpha_1 & \sqrt{\beta_2} & \dots & 0 \\ 0 & \sqrt{\beta_2} & \alpha_2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \alpha_{n-1} \end{pmatrix}$$

The eigenvalues of J give us the quadrature points; the weights are computed from the first components of the normalized eigenvectors.

Because J is symmetric and tridiagonal, the computation is not difficult, so it might seem we are all set...

PLAN: Truncated Normal is a Problem

Unfortunately, the Hermite polynomials $H_i(x)$ are the correct orthogonal family for the normal distribution, but **not** for the truncated normal distribution;

I don't know the family for the truncated distribution.

And remember, we say "the" truncated normal distribution, but every choice of the parameters $\overline{\mu}, \overline{\sigma}, a, b$ really gives us a completely different distribution, and presumably a different orthogonal family to work out...if we could work it out.

Fortunately, the orthogonal family method is not the only way.

Golub and Welsch described an alternative, the moment matrix method.

PLAN: Quadrature by Moment Matrix

The k-th moment of a PDF is simply the value of the integral

$$m_k = \int_a^b x^k \, \text{pdf}(x) \, dx$$

Then, to compute an n-point Gauss quadrature rule for the PDF, you construct the $(n+1)x(n+1)$ moment matrix:

$$M = \begin{pmatrix} m_0 & m_1 & m_2 & \dots & m_n \\ m_1 & m_2 & m_3 & \dots & m_{n+1} \\ m_2 & m_3 & m_4 & \dots & m_{n+2} \\ \dots & \dots & \dots & \dots & \dots \\ m_n & m_{n+1} & m_{n+2} & \dots & m_{2n} \\ \end{pmatrix}$$

We compute the upper Cholesky factorization $M = R'R$. From the entries of R it is possible to compute the vectors α and β needed to construct the Jacobi matrix J and hence to determine the points and weights of the quadrature rule.

PLAN: So Now What's Missing?

A quadrature rule is within our grasp, if only we can compute moments.

To compute the k-th moment m_k of the truncated normal distribution, we compute:

$$m_k = \frac{1}{S\sqrt{2\pi\overline{\sigma}^2}} \int_a^b x^k e^{-\frac{(x-\overline{\mu})^2}{2\overline{\sigma}^2}} \, dx$$

Evaluating this integral is not trivial.

But when we see a hard integral, isn't Mathematica always the solution?

...Not necessarily...Unless Mathematica can give me a usable formula for the answer, I'll have to precompute m_k for every reasonable choice of a, b and k, and those are the only values my program will work with.

So let's just see how helpful Mathematica will be.

PLAN: Mathematica Tries to Help

To keep things simple, let's just ask for the moments of the original un-truncated normal distribution:

```
    In[1] = 1/(S Sqrt[2 Pi s^2]
      Integrate [ x^k Exp[-(x - m)^2/(2 s^2)],
      {x, -Infinity, +Infinity}]
```

```
Out[1] = ConditionalExpression[(1/(S Sqrt[\[Pi]] s))
  2^(-(1/2) + 1/2 (-1 + k)) E^(-(m^2/(2 s^2))) (1/s^2)^(
  1/2 (-1 - k)) (-Sqrt[2] (-1 + (-1)^k) m Sqrt[1/s^2]
  Gamma[1 + k/2] Hypergeometric1F1[1 + k/2, 3/2, m^2/(
  2 s^2)] + (1 + (-1)^k) Gamma[(1 + k)/2] Hypergeometric1F1[(
  1 + k)/2, 1/2, m^2/(2 s^2)]), Re[1/s^2] > 0 && Re[k] > -1]
```

As they say about getting answers from mathematicians, this response is totally correct and totally useless! I do not want to study the Mathematica manual and a special functions handbook to find out what this means and whether I can use it.

Is there any other way to find the moments m_k?

PLAN: Maybe the Internet isn't entirely bad

Since our library was closed for six weeks, in order to throw out most of the books, and hide the others, I decided to try to search on the internet for a useful formula for the moments of the normal distribution. Since the words **moment** and **normal** occur in many contexts, this was not so easy.

But I came across a wonderful page by John D. Cook, titled *General formula for normal moments*:

$$m_k = E\{x^k\} = E\{(\sigma\xi + \mu)^k\} = \sum_{i=0}^{k} \binom{k}{i} E(\xi^i)\,\sigma^i\,\mu^{k-i}$$
$$= \sum_{j=0}^{\lfloor k/2 \rfloor} \binom{k}{2j} (2j-1)!!\,\sigma^{2j}\,\mu^{k-2j}$$

Here,

- $\lfloor k/2 \rfloor$ is the *floor()* function, which rounds down to an integer;
- $(2j-1)!!$ is using the *double factorial function* which here is simply the product of the odd numbers from 1 to $2j-1$;
- $\binom{k}{2j}$ is the combinatorial coefficient, here $\frac{k!}{(2j)!(k-2j)!}$.

PLAN: We're Almost There

That gives us the moments for the standard normal distribution.

Actually, I still need the moments of the truncated normal distribution.

But if I can get those moments m_k, I really am done, because:

- I can construct the moment matrix M,
- compute the Cholesky factor R,
- determine the vectors α and β,
- built the Jacobi matrix J,
- get the eigenvalues and eigenvectors,
- and finally...get the quadrature rule x and w.

So after all these clues, we might be almost ready to make an arrest!

Truncated Normal Collocation

- The Strange Case of the Abnormal Normal
- Can You Describe the Suspect?
- I Have a Cunning Plan
- **A Matter of Moments**
- The Summing Up

MOMENTS: The Mysterious Phoebus J Dhrymes

Returning to the Internet, I ran across a reference in a paper to an unpublished note (not a paper) by Phoebus J Dhrymes, titled *Moments of Truncated (Normal) Distributions*. I was able to find a web site for this combination statistician/psychiatrist, and indeed, there was a 3-page discussion that included the "one-armed truncation" formula:

$$E(x^k|x \le b) = \frac{1}{S\sqrt{2\pi\overline{\sigma}^2}} \int_{-\infty}^{b} x^k e^{-\frac{(x-\overline{\mu})^2}{2\overline{\sigma}^2}} dx = \sum_{i=0}^{k} \binom{k}{i} \sigma^i \mu^{k-i} L_i$$

Recall $\phi(x)$ and $\Phi(x)$ are the PDF and CDF, of the standard normal distribution and define:

$$\beta = \frac{b-\overline{\mu}}{\overline{\sigma}}$$

Then the quantity L_i satisfies the recursion:

$$\begin{aligned} L_0 =& 1 \\ L_1 =& -\frac{\phi(\beta)}{\Phi(\beta)} \\ L_i =& -\beta^{i-1}\frac{\phi(\beta)}{\Phi(\beta)} + (i-1)L_{i-2} \end{aligned}$$

MOMENTS: Upper Truncated Normal Moments!

So now I had a formula for the moments of the upper truncated normal distribution! I programmed it, and as a test, I asked Mathematica to compute the corresponding integral for specific choices of the exponent, the normal parameters μ and σ, and the upper truncation limit b;

For example, for $\overline{\mu} = 5$, $\overline{\sigma} = 1$, $b = 10$:

Order	Moment	Mathematica
0	1	1
1	5	5
2	26	26
3	140	140
4	777.997	777.997
5	4449.97	4449.97
6	26139.69	26139.67
7	157396.75	157396.71
8	969946.73	969946.45

These results gave me some confidence that Dhrymes's formula was correct, and implemented correctly.

MOMENTS: Lower Truncated Normal Moments!

I soon realized that the formula for upper truncated moments also gave me the formula for lower truncated moments, since, by a change of variable $y = -x$, we can get:

$$
\begin{aligned}
m_k &= \frac{1}{S\sqrt{2\pi\overline{\sigma}^2}} \int_a^\infty x^k e^{-\frac{(x-\overline{\mu})^2}{2\overline{\sigma}^2}}\, dx \\
&= \frac{1}{S\sqrt{2\pi\overline{\sigma}^2}} \int_a^\infty (-y)^k e^{-\frac{(-y-\overline{\mu})^2}{2\overline{\sigma}^2}}\, dx \\
&= \frac{(-1)^k}{S\sqrt{2\pi\overline{\sigma}^2}} \int_{-\infty}^{-a} y^k e^{-\frac{(y--\overline{\mu})^2}{2\overline{\sigma}^2}}\, dy
\end{aligned}
$$

or ± 1 times the k-th upper truncated normal moment for $-\overline{\mu}$ and $\overline{\sigma}$, with $-a$ as the upper limit.

So I had moment formulas for the normal, lower truncated, and upper truncated distributions, but nothing on the doubly truncated distribution!

MOMENTS: Double Truncated Normal Moments!

At last, I found a paper online that referenced Phoebus J Dhrymes, and stated that Dhrymes's result also implied a simple formula for moments of the doubly truncated normal distribution.

Define

$$\alpha = \frac{a - \overline{\mu}}{\overline{\sigma}}; \quad \beta = \frac{b - \overline{\mu}}{\overline{\sigma}}$$

Then (as before) we have

$$m_k = \sum_{i=0}^{k} \binom{k}{i} \overline{\sigma}^i \overline{\mu}^{k-i} L_i$$

where the quantity L_i satisfies the recursion:

$$\begin{aligned} L_0 &= 1 \\ L_1 &= -\frac{\phi(\beta) - \phi(\alpha)}{\Phi(\beta) - \Phi(\alpha)} \\ L_i &= -\frac{\beta^{i-1}\phi(\beta) - \alpha^{i-1}\phi(\alpha)}{\Phi(\beta) - \Phi(\alpha)} + (i-1)L_{i-2} \end{aligned}$$

MOMENTS: Double Truncated Normal Moments!

Now that I had usable moment formulas for all four cases, I added a seventh "task", to compute the k-th moment, to the **truncated_normal** library.

And I was now ready to consider the next step, which was to implement the moment formulation of the Golub-Welsch algorithm for computing quadrature rules.

http://people.sc.fsu.edu/∼jburkardt/m_src/truncated_normal/truncated_normal.html

Truncated Normal Collocation

- The Strange Case of the Abnormal Normal
- Can You Describe the Suspect?
- I Have a Cunning Plan
- A Matter of Moments
- **The Summing Up**

SUMUP: One Step Beyond

We are now ready to try to compute an *n*-point quadrature rule using the moment-based version of the Golub Welsch algorithm - that is, once we write a program to implement the Golub Welsch algorithm!

The first step of the Golub Welsch algorithm requires us to form the order $n+1$ moment matrix M, filling it with the values of moments m_0 through m_{2n}, which we just figured out how to compute.

The second step requires us to compute the upper Cholesky factor R such that $M = R'R$ - but it's not difficult to put together the code for this calculation either.

Golub and Welsch then supply formulas for extracting the vectors α and β from the information in R, and with these, we can construct the *nxn* symmetric tridiagonal Jacobi matrix J.

Now comes the tricky step - compute eigenvalues and eigenvectors of J.

At this point, you might respond - *That's easy, just use Matlab!*

SUMUP: Jacobi will get me the eigenvalues

I want my code to be accessible in several languages. So I really want to write down an eigenvalue routine explicitly. Luckily, our symmetric matrix J is ideal for the Jacobi eigenvalue algorithm.

The basic idea of the Jacobi algorithm is to pre- and post-multiply the matrix by Jacobi rotation matrices that zero out the largest off-diagonal element. As this process is repeated, the matrix rapidly approximates a diagonal matrix, from which the eigenvalues can be read off.

Moreover, I had forgotten that this algorithm can also return the corresponding eigenvectors, which we need to have for the quadrature weights.

So my next task was to write a library called **jacobi_eigenvalue** which would allow me to test my implementation on some standard problems.

http://people.sc.fsu.edu/∼jburkardt/m_src/jacobi_eigenvalue/jacobi_eigenvalue.html

SUMUP: The Golub-Welsch Moment Algorithm Executes

Once the Jacobi eigenvalue algorithm was working, I had all the pieces needed to carry out a Golub-Welsch momentum algorithm.

Since I had never done this before, I first set up momentum calculations for Legendre, Laguerre, and Normal distributions, because I knew what the associated quadrature rules should be in these cases, and I caught some small errors this way.

Then I added in the momentum calculations for the truncated normal distribution, and was finally able to compute some example rules.

I called this hunk of software **quadmom** for *quadrature by moments*.

http://people.sc.fsu.edu/∼jburkardt/m_src/quadmom/quadmom.html

SUMUP: An Interactive Rule Calculator

Using **quadmom** requires writing and compiling a calling program.

A more convenient approach is an executable program, **truncated_normal_rule**, that only asks for input:

- *option* 0/1/2/3 for none, lower, upper, double truncation;
- *n* the number of points in the rule;
- *mu*, the mean of the original normal distribution;
- *sigma* the standard deviation of the original normal distribution;
- *a* the left endpoint (for options 1 or 3);
- *b* the right endpoint (for options 2 or 3);
- *filename*, the root name of the output files.

http://people.sc.fsu.edu/∼jburkardt/m_src/truncated_normal_rule/truncated_normal_rule.html

SUMUP: An Interactive Rule Calculator

To compute a doubly truncated quadrature rule of 10 points, with $\overline{\mu} = 0$ and $\overline{\sigma} = 1$, over the interval $[-3.0, +3.0]$, we write:

```
truncated_normal_rule 3 10 0.0 1.0 -3.0 +3.0 double10
```

For a lower truncated rule over $[-3.0, \infty)$, write:

```
truncated_normal_rule 1 10 0.0 1.0 -3.0 lower10
```

Dropping both limits, we get a non-truncated normal rule:

```
truncated_normal_rule 0 10 0.0 1.0 normal10
```

SUMUP: A Sample Rule Computation

The program writes the rule to three output files, containing the points, the weights, and the integration limits. Thus, the lower truncated normal rule we requested above would be stored in the following files:

```
lower10_x.txt      lower10_w.txt       lower10_r.txt

-2.83915            0.00279            -3.00000
-2.28008            0.02023             1.0E+30  <-- ``Infinity'
-1.53530            0.09714
-0.71994            0.25745
 0.12958            0.34188
 1.00659            0.21473
 1.91770            0.05925
 2.88043            0.00629
 3.93128            0.00019
 5.16662            0.8E-06
```

(We actually write more digits in the real files.)

SUMUP: Testing the Rule

Let's stick with the lower truncated normal distribution, with parameters $\overline{\mu} = 0$ and $\overline{\sigma} = 1$, over the interval $[-3.0, \infty)$, and estimate the following integral using an *n*-point rule.

$$Q = \frac{1}{S\sqrt{2\pi\overline{\sigma}^2}} \int_a^{+\infty} \sin(x) e^{-\frac{(x-\overline{\mu})^2}{2\overline{\sigma}^2}} \, dx$$

N	Estimated Q
1	0.004437820
2	-0.002956940
3	0.000399622
4	-0.000236540
5	-0.000173932
6	-0.000177684
7	-0.000177529
8	-0.000177534
9	-0.000177534
Mathematica	-0.000177531

SUMUP: Multivariate Integration

Any quadrature rule for a one-dimensional interval $[a, b]$ can be used to construct a product rule for the region $[a, b] \times [a, b]$. Such a product rule can be used to estimate corresponding integrals of a function $f(x, y)$ over the product region.

If the original rule used n points with coordinates x_i and weights w_i, the new rule will use n^2 points, consisting of all possible pairings of two x values, weighted by the product of the corresponding w values. For example, a simple three point rule for [0,1] might be:

W	X
$1/4$	0.0
$1/2$	0.5
$1/4$	1.0

SUMUP: Multivariate Integration

The corresponding product rule for $[0,1]^2$ would then be:

W	X	Y
1/16	0.0	0.0
1/8	0.5	0.0
1/16	1.0	0.0
1/8	0.0	0.5
1/4	0.5	0.5
1/8	1.0	0.5
1/16	0.0	1.0
1/8	0.5	1.0
1/16	1.0	1.0

The procedure is flexible, and allows us to shift or scale the interval in any dimension, to use a different number of points with a given rule, or to use completely different rules in each dimension as appropriate.

The main point is, however, that this means that once we have developed a procedure for producting one dimensional quadrature rules for the truncated normal distribution, we have automatically a procedure for generating similar rules for problems in higher dimensions.

SUMUP: Sparse Grids

If we are interested in multivariate integration, then technically the product rule approach gives us everything we need. However, notice that the number of points used by a product rule involves raising the number of points in the 1D rule to the power of the dimension. While a 3 point rule in 1D looks cheap, in 10 dimenension it needs almost 60,000 points and more than 3 billion points in 20 dimensions.

When modeling stochastic problems, each random variable corresponds to a dimension, and it's not uncommon for users to want 40 or 50 such variables. This would seem to rule out the use of standard quadrature rules, leaving nothing but Monte Carlo sampling to estimate the integrals.

SUMUP: Sparse Grids

Luckily, we know how to randomly sample from the truncated normal distribution. But if we wish to take advantage of the much improved convergence rate of quadrature rules, we can explore the construction of a sparse grid based on our 1D quadrature rules.

In the case of a simple 3 point rule, a sparse grid will use 31 points in 10 dimensions, 61 points in 20 dimensions, and 121 points in 40 dimensions. Although even a sparse grid will run into limitations as the 1D rule increases, it is able to efficiently give integral estimates of known preicision in dimensions that are unimaginable for the product rule.

A sparse grid is easily constructed by combining simple product grids; since we know how to do that for the truncated normal distribution, it's just of a question of working out the rule for puting them together that would remain.

CONCLUSION:

I hope I've given you an idea of the kinds of problems I look at, and how I go about trying to solve them,

The main moral I can give is that, in a scientific computing project, you are almost always "an infinite distance away" from your final, happy working code, and so you have to just pay very close attention to what you are doing right now, making sure you have understood what you need to do, how it fits into the big picture, and why you can demonstrate that it is correct.