1. The normal-distribution $\mathbf{N}(\mu, \sigma^2)$. The normal-distribution is one of the important distributions in statistics. As we know, it is sufficient to concentrate on the standard normal distribution $\mathrm{N}(\mu = 0, \sigma^2 = 1)$, which has the density function:

$$\phi(z) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{1}{2} z^2\right) I_{(-\infty,+\infty)}(z)$$

This paper describes two *APL* functions for the density function (DSNV) and the distribution function (VSNV) of the standard normal.

2. The density of the standard normal distribution can be programmed directly:

```
∇ R←DSNV Z   2
R←(⋆¯0.5×Z×Z)×(○2)⋆¯0.5
```

See also section 21.

DSNV appears in section 1.

3. The calculation of the distribution function is not such an easy task. Following Kennedy/Gentle we concentrate on the approximation introduced by Cody which is based on rational Tschebychev approximations for the error functions $\mathrm{erf}(z)$ and $\mathrm{erfc}(z)$.

$$\begin{aligned}
\mathrm{erf}(z) &= \frac{2}{\sqrt{\pi}} \int_0^z \exp(-t^2) dt \quad , z > 0 \\
\mathrm{erfc}(z) &= 1 - \mathrm{erf}(z) \\
\mathrm{erf}(-z) &= -\mathrm{erf}(z) \\
\Phi(z) &= \begin{cases} \frac{1}{2}\{1 + \mathrm{erf}(u)\} & \text{if } u \geq 0 \\ \frac{1}{2}\mathrm{erfc}(-u) & \text{if } u < 0 \end{cases} \quad \text{with } u = z/\sqrt{2}
\end{aligned}$$

To avoid loss of significance one should use the approximation for $\mathrm{erf}(z)$ only for those z for which $\mathrm{erf}(z) < \mathrm{erfc}(z)$ (see Hart p.138). $\mathrm{erf}(z)$ is then computed according to $\mathrm{erfc}(z)$=1-$\mathrm{erf}(z)$. In all other cases one should approximate $\mathrm{erfc}(z)$ first and then use the formula just mentioned to get $\mathrm{erf}(z)$. Cody suggests $z = 0.46875$ to be the critical value. A brief look at table 7.1 in Abramowitz/Stegun shows that this does not make sense. The intersection of both function is $z \approx 0.476875$, so Cody's value is probably a misprint. We will take the given point of intersection instead. Cody's approximations are of the form:

$$\begin{aligned}
\mathrm{erf}(z) &\approx z R_{lm}(z^2) && \text{if } 0 < z \leq 0.5 \\
\mathrm{erfc}(z) &\approx \exp(-z^2) R_{lm}(z) && \text{if } 0.476875 \leq z \leq 4 \\
\mathrm{erfc}(z) &\approx \tfrac{1}{z} \exp(-z^2) \{ \tfrac{1}{\sqrt{\pi}} + \tfrac{1}{z^2} R_{lm}(z^{-2}) \} && \text{if } 4 \leq z
\end{aligned}$$

Cody provides R_{lm} for various polynomial orders together with maximal relative errors. We have chosen the polynomial fractions of the orders (3,3), (7,7) and (4,4) for the three intervals. The maximal error is according to Cody smaller than 2.3×10^{-15}. This seems to be appropriate with respect to the accuracy of our machine.

One may say, that a disadvantage of this approximation is the large number of coefficients, which enlarge the program code. Truely one can get a very short code using other approximations such as those given by Hill, Adams, Hastings, Andrews and Wheeler.

But as experiments made by Steinecker showed, Cody's approximation is superior to all the others, including Wheeler's PGAUSS, in respect to speed and accuracy, we will implement Cody's Algorithm as the *APL*-function VSNV.

Some of the other suggestions will be dealt with later.

4. So let us start with Cody's approximation for the normal distribution function. We may write down the function at a higher level as follows.

∇ P←VSNV Z 4
⟨ Do the initialisation steps. 5 ⟩
⟨ Jump to L1, if there are no small z-values. 11 ⟩
⟨ Do Cody's approximation for small z-values. 12 ⟩
L1:
⟨ Jump to L2, if there are no medium z-values. 15 ⟩
⟨ Do Cody's approximation for medium z-values. 16 ⟩
L2:
⟨ Jump to L3, if there are no large z-values. 17 ⟩
⟨ Do Cody's approximation for large z-values. 18 ⟩
L3:
⟨ Do the final adjustments. 19 ⟩

See also section 22.
VSNV appears in sections 1 and 3.

5. The initialisation process consists of

- transform z
- classify z
- allow for multiple z-values

So the initialisation process can be refined as follows
⟨ Do the initialisation steps. 5 ⟩ ≡

⟨ Transform z, mark negative z-values, switch to absolute values. 6 ⟩
⟨ Mark large z-values. 7 ⟩
⟨ Mark medium z-values. 8 ⟩
⟨ Prepare for multiple input. 9 ⟩
This code is used in section 4.

6. Let us continue with stepwise refinement. First we have to do $u = z/\sqrt{2}$ and then to decide wether u is less or equal than 0 or not. We use a local binary variable `un` to store wether the corresponding element of u is less or equal 0 or not. Then we take the absolute values of z which will be stored in `z`. So from now on we should take into consideration that `z` contains the absolute values of the input variable `Z`. We also initialize the output variable `P` of our function.

⟨ Transform z, mark negative z-values, switch to absolute values. 6 ⟩ ≡

```
P←un←0≥z←,Z÷2⋆0.5 ⋄ z←|z
```

This code is used in section 5.

7. Now we mark those z-values which are large, i.e. where z is greater than 4 by storing their indices in `z` in the local variable `i3`. Futhermore we store all indices of `z` in the local variable `i1` and keep the marks of those elements which are greater than 4 in `ul`.

⟨ Mark large z-values. 7 ⟩ ≡

```
i3←(ul←z>4)/i1←ιρz
```

This code is used in section 5.

8. Medium z-values are those where z is greater than 0.476875 and not greater than 4. We keep their indices in the local variable `i2`. We can use the fact, that the binary variable `ul` contains already the information which values are greater than 4. But first we modify `i1` so that `i1` contains all elements of `z` which are not large, i.e. where z is not greater than 4. Besides that we introduce the variable `um` which contains the information wether `z[i1]` is greater 0.476875 or not.

⟨ Mark medium z-values. 8 ⟩ ≡

```
i2←(um←z[i1]>0.476875)/i1←(~ul)/i1
```

This code is used in section 5.

9. Now we form `z` into a matrix because we want to deal with arrays as right argument. We provide a matrix of z^2 too.

⟨ Prepare for multiple input. 9 ⟩ ≡

```
z2←,zz2←zz×zz←((ρz),1)ρz
```

This code is used in section 5.

10. We should make some variables local. We introduce the convention that local variables will be written in lower case letters

⟨ Local Variables of VSNV(4) 10 ⟩ ≡

```
i1,i2,i3,un,um,ul,z2,zz2,z,zz
```

See also section 20.

11. If there are no small z-values we carry on our computation at the label L1.

⟨ Jump to L1, if there are no small z-values. 11 ⟩ ≡

```
→(0=ρi1←(~um)/i1)/L1
```

This code is used in section 4.

12. To evaluate Cody's approximation in all cases, we first set the coefficients and then compute the result. This will become explicit when we handle the case of small values.

⟨ Do Cody's approximation for small z-values. 12 ⟩ ≡
⟨ Set the coefficients for the small value approximation. 13 ⟩
⟨ Compute P for small z-values. 14 ⟩

This code is used in section 4.

13. As was said before the rational function used is of order $l = 3$, $m = 3$. The coefficients of the respective polynomials are stored in a and b.

⟨ Set the coefficients for the small value approximation. 13 ⟩ ≡

```
a←¯0.035609843701815386 6.9963834886191355 21.979261618294153
    242.66795523053176
b←1 15.082797630407788 91.1649054045149 215.0588758698612
```

This code is used in section 12.

14. The approximation is done according to the following formulas.

$$\begin{aligned}\Phi(z) &= \frac{1}{2}\{1 + \operatorname{erf}(z/\sqrt{2})\} \\ \operatorname{erf}(z) &\approx zR_{lm}(z^2)\end{aligned}$$

⟨ Compute P for small z-values. 14 ⟩ ≡

```
P[i1]←0.5×1+z[i1]×(zz2[i1;]⊥a)÷zz2[i1;]⊥b
```

This code is used in section 12.

15. ⟨ Jump to L2, if there are no medium z-values. 15 ⟩ ≡

```
→(0=ρi2)/L2
```

This code is used in section 4.

16. The further computatation goes on in the same manner. First we set some coefficients, then we compute the corresponding P. Now the rational function used is of order $l = 7$, $m = 7$. Again the coefficients of the respective polynomials are stored in a and b. The approximation is done according to the following formulas.

$$\begin{aligned} \Phi(z) &= \frac{1}{2}\{2 - \operatorname{erfc}(z/\sqrt{2})\} \\ \operatorname{erfc}(z) &\approx \exp(-z^2) R_{lm}(z) \end{aligned}$$

⟨ Do Cody's approximation for medium z-values. 16 ⟩ ≡

```
a←¯1.3686485738271671E¯7 0.564195517478974 7.2117582508830935
    43.162227222056735 152.98928504694041 339.3208167343437
    451.91895371187293 300.45926102016161
b←1 12.782727319629473 77.00015293522947 277.58544474398764
    638.98026446563124 931.35409485060966 790.95092532789811
    300.45926095698333
P[i2]←1−0.5×(⋆−z2[i2])×(zz[i2;]⊥a)÷zz[i2;]⊥b
```

This code is used in section 4.

17. ⟨ Jump to L3, if there are no large z-values. 17 ⟩ ≡

```
→(0=ρi3)/L3
```

This code is used in section 4.

18. At last we compute P for the large z-values. The scheme is as before. First we set some coefficients, then we compute the corresponding P. Here the rational function used is of order $l = 4$, $m = 4$. Again the coefficients of the respective polynomials are stored in a and b. The approximation is done according to the following formulas.

$$\begin{aligned} \Phi(z) &= \frac{1}{2}\{2 - \operatorname{erfc}(z/\sqrt{2})\} \\ \operatorname{erfc}(z) &\approx \frac{1}{z}\exp(-z^2)\{\frac{1}{\sqrt{\pi}} + \frac{1}{z^2} R_{lm}(z^{-2})\} \end{aligned}$$

⟨ Do Cody's approximation for large z-values. 18 ⟩ ≡

```
a←¯0.02231924597341847 ¯0.2786613086096478 ¯0.22695659353968692
    ¯0.04947309106232507 5 ¯2.9961070770354215E¯3
```

```
b←1 1.9873320181713526 1.051675107067932 0.19130892610782985
    0.010620923052846794
h←(÷(○1)⋆0.5)+(÷z2[i3])×(h⊥a)÷(h←÷zz2[i3;])⊥b
P[i3]←1−0.5×((⋆−z2[i3])÷z[i3])×h
```

This code is used in section 4.

19. Finally, we must adjust P for negative input values, for we have set z to its magnitude. We apply $F(z) = 1 - F(z)$. Our binary variable un contains the information wether the z was greater 0 or not. So we only have to substract P from un and take the magnitude again to get the sign right again for positive values.

⟨ Do the final adjustments. 19 ⟩ ≡

```
P←|un−P
```

This code is used in section 4.

20. We should make some more variables local.

⟨ Local Variables of VSNV(4) 10 ⟩ + ≡

```
h,a,b
```

21. The *APL*-functions.

```
∇ R←DSNV Z   2
⍝  Initially declared in WEB-file <norm1.awb>, line 18; date: Tue Sep 29
     12:12:43 1992
⍝  Main WEB-file <work.awb>
⍝  ∇DSNV: 2
⍝  2:
R←(*¯0.5×Z×Z)×(○2)*¯0.5
⍝  :2
```

22.

```
∇ P←VSNV Z   4
⍝  Initially declared in WEB-file <norm1.awb>, line 82; date: Tue Sep 29
     12:12:43 1992
⍝  Main WEB-file <work.awb>
⍝  ∇VSNV: 4
⍝  10: i1,i2,i3,un,um,ul,z2,zz2,z,zz
⍝  20: h,a,b
⍝  4:, 5:, 6:
P←un←0≥z←,Z÷2*0.5 ⋄ z←|z
⍝  :6, 7:
i3←(ul←z>4)/i1←⍳ρz
⍝  :7, 8:
i2←(um←z[i1]>0.476875)/i1←(~ul)/i1
⍝  :8, 9:
z2←,zz2←zz×zz←((ρz),1)ρz
⍝  :9, :5, 11:
→(0=ρi1←(~um)/i1)/L1
⍝  :11, 12:, 13:
a←¯0.035609843701815386 6.9963834886191355 21.979261618294153
     242.66795523053176
b←1 15.082797630407788 91.1649054045149 215.0588758698612
⍝  :13, 14:
P[i1]←0.5×1+z[i1]×(zz2[i1;]⊥a)÷zz2[i1;]⊥b
⍝  :14, :12
L1:
⍝  15:
→(0=ρi2)/L2
⍝  :15, 16:
```

```
 a←¯1.3686485738271671E¯7 0.564195517478974 7.2117582508830935
     43.162227222056735 152.98928504694041 339.3208167343437
     451.91895371187293 300.45926102016161
 b←1 12.782727319629473 77.00015293522947 277.58544474398764
     638.98026446563124 931.35409485060966 790.95092532789811
     300.45926095698333
 P[i2]←1−0.5×(⋆−z2[i2])×(zz[i2;]⊥a)÷zz[i2;]⊥b
⍝  :16
L2:
⍝  17:
 →(0=ρi3)/L3
⍝  :17, 18:
 a←¯0.02231924597341847 ¯0.2786613086096478 ¯0.22695659353968692
     ¯0.049473091062325075 ¯2.9961070770354215E¯3
 b←1 1.9873320181713526 1.051675107067932 0.19130892610782985
     0.010620923052846794
 h←(÷(○1)⋆0.5)+(÷z2[i3])×(h⊥a)÷(h←÷zz2[i3;])⊥b
 P[i3]←1−0.5×((⋆−z2[i3])÷z[i3])×h
⍝  :18
L3:
⍝  19:
 P←|un−P
⍝  :19, :4
 T0_expand
⍝  Main WEB-file: work.awb
```