

In and Out

Edited by Paul Abbott

In and Out offers readers an opportunity to ask questions of the experts. *The Mathematica Journal* encourages readers to submit problems in care of the editor. Answers posted to the *Mathematica* newsgroup, comp.soft-sys.math.mathematica, that appear here are edited for clarity and length.

■ Closed Form Integral

In the last column, a closed form for the integral

$$a_{k,l} = \int_0^1 P_k(x) P_l(2x-1) dx$$

was obtained. Here was the result, with the denominator simplified slightly by noting that $(2l+1)(1)_l(l+1)_l = (2l+1)!$.

$$a_{k,l} = \frac{1}{(2l+1)!} ((-1)^l 2^{-k} {}_2F_1(l-k, -k+l+1; 2l+2; -1) (-k)_l (k+1)_l);$$

Elmar Zeitler (zeitler@fhi-berlin.mpg.de) suggested an alternative approach: expanding $P_k(x)$ as a Gaussian hypergeometric function (functions.wolfram.com/Polynomials/LegendreP/26/01/01/0001),

$$P_k(x) = {}_2F_1\left(-k, k+1; 1; \frac{1-x}{2}\right),$$

and using functions.wolfram.com/05.03.07.0004.01 to write

$$P_l(2x-1) = \frac{1}{l!} \frac{\partial^l (x^2-x)^l}{\partial x^l},$$

one has to compute

$$\int_0^1 (1-x)^m \frac{1}{l!} \frac{\partial^l (x^2-x)^l}{\partial x^l} dx,$$

which can be done using integration by parts. Alternatively, evaluating the integral for $l = 0, 1, \dots, 4$, the pattern is clear.

In[2]:= Simplify[Table[{l, $\int_0^1 (1-x)^m \frac{1}{l!} \frac{\partial^l (x^2-x)^l}{\partial x^l} dx$ }, {l, 0, 4}], m ≥ 0] // Factor

$$\text{Out[2]} = \begin{pmatrix} 0 & \frac{1}{m+1} \\ 1 & -\frac{m}{(m+1)(m+2)} \\ 2 & \frac{(m-1)m}{(m+1)(m+2)(m+3)} \\ 3 & -\frac{(m-2)(m-1)m}{(m+1)(m+2)(m+3)(m+4)} \\ 4 & \frac{(m-3)(m-2)(m-1)m}{(m+1)(m+2)(m+3)(m+4)(m+5)} \end{pmatrix}$$

One observes that

$$\int_0^1 (1-x)^m \frac{1}{l!} \frac{\partial^l (x^2-x)^l}{\partial x^l} dx = \frac{(-m)_l}{(m+1)_{l+1}},$$

leading to an alternative form for $a_{k,l}$, denoted $b_{k,l}$.

In[3]:= $b_{k,l} = \sum_{m=0}^k \frac{(-k)_m (k+1)_m}{(1)_m m! 2^m} \frac{(-m)_l}{(m+1)_{l+1}}$ // FullSimplify

$$\text{Out[3]} = (-1)^l {}_3\tilde{F}_2\left(1, -k, k+1; 1-l, l+2; \frac{1}{2}\right)$$

Since $a_{k,l} = 0$ for $l > k$, we check that $a_{k,l} = b_{k,l}$ for $0 \leq l \leq k \leq 4$.

In[4]:= Table[{k, l, $a_{k,l} = b_{k,l}$ }, {k, 0, 4}, {l, 0, k}]

$$\text{Out[4]} = \left\{ (0 \ 0 \ \text{True}), \begin{pmatrix} 1 & 0 & \text{True} \\ 1 & 1 & \text{True} \end{pmatrix}, \begin{pmatrix} 2 & 0 & \text{True} \\ 2 & 1 & \text{True} \\ 2 & 2 & \text{True} \end{pmatrix}, \begin{pmatrix} 3 & 0 & \text{True} \\ 3 & 1 & \text{True} \\ 3 & 2 & \text{True} \\ 3 & 3 & \text{True} \end{pmatrix}, \begin{pmatrix} 4 & 0 & \text{True} \\ 4 & 1 & \text{True} \\ 4 & 2 & \text{True} \\ 4 & 3 & \text{True} \\ 4 & 4 & \text{True} \end{pmatrix} \right\}$$

Alternatively, expanding both $P_k(x)$ and $P_l(2x-1) = (-1)^l P_l(1-2x)$ as Gaussian hypergeometric functions, the result is immediate.

In[5]:= $\sum_{m=0}^k \frac{(-k)_m (k+1)_m}{(1)_m m! 2^m} \sum_{n=0}^l \frac{(-l)_n (l+1)_n}{(1)_n n!}$

Assuming[$m \geq 0 \wedge n \geq 0$, $(-1)^l \int_0^1 (1-x)^m x^n dx$] // FullSimplify

$$\text{Out[5]} = (-1)^l {}_3\tilde{F}_2\left(1, -k, k+1; 1-l, l+2; \frac{1}{2}\right)$$

In other words, we have demonstrated the identity

$$\frac{2^{-k} {}_2F_1(l-k, -k+l+1; 2l+2; -1) (-k)_l (k+1)_l}{(2l+1)!} = {}_3\tilde{F}_2\left(1, -k, k+1; 1-l, l+2; \frac{1}{2}\right).$$

■ Sum of Gaussians

Q: I would like to find a closed form for the sum

$$\text{In}[1]:= f(z) = \sum_{n=-\infty}^{\infty} e^{-\frac{1}{2}(z+n)^2}$$

$$\text{Out}[1]= \sum_{n=-\infty}^{\infty} e^{-\frac{1}{2}(n+z)^2}$$

Clearly $f(z)$ is periodic with unit period (put $z \rightarrow z + 1$ and $n \rightarrow n - 1$). How can I compute the Fourier series expansion of $f(z)$?

A: Putting $\tau \rightarrow 2i\pi$ and $z \rightarrow \pi z$ into the formula functions.wolfram.com/09.03.06.0019.01,

$$\vartheta_3(z, q) = \frac{\sqrt{i}}{\sqrt{\tau}} \sum_{n=-\infty}^{\infty} e^{-\frac{\pi i}{\tau} (n + \frac{z}{\pi})^2} \quad ; \quad q = e^{i\pi\tau},$$

that is,

$$\text{In}[2]:= \left\{ \frac{\sqrt{i}}{\sqrt{\tau}} e^{-\frac{\pi i}{\tau} (n + \frac{z}{\pi})^2}, q = e^{i\pi\tau} \right\} /. \{\tau \rightarrow 2i\pi, z \rightarrow \pi z\} // \text{FullSimplify}$$

$$\text{Out}[2]= \left\{ \frac{e^{-\frac{1}{2}(n+z)^2}}{\sqrt{2\pi}}, q = e^{-2\pi^2} \right\}$$

we can express the sum in terms of **EllipticTheta** as follows:

$$\sum_{n=-\infty}^{\infty} e^{-\frac{1}{2}(n+z)^2} = \sqrt{2\pi} \vartheta_3(\pi z, e^{-2\pi^2}).$$

The Fourier series expansion of $\vartheta_3(z, q)$ (essentially its standard definition) is given at functions.wolfram.com/09.03.06.0001.01,

$$\vartheta_3(z, q) = 1 + 2 \sum_{k=1}^{\infty} q^{k^2} \cos(2kz) \quad ; \quad |q| < 1.$$

Here are the first few terms of the Fourier series expansion of $f(z)$.

$$\text{In}[3]:= \sqrt{2\pi} \left(1 + 2 \sum_{k=1}^3 q^{k^2} \cos(2kz) \right) /. \{z \rightarrow \pi z, q \rightarrow e^{-2\pi^2}\}$$

$$\text{Out}[3]= \sqrt{2\pi} (2(e^{-2\pi^2} \cos(2\pi z) + e^{-8\pi^2} \cos(4\pi z) + e^{-18\pi^2} \cos(6\pi z)) + 1)$$

Note that the Fourier coefficients decrease extremely rapidly.

```
In[4]:= q^Range[3]^2 /. q -> e^{-2\pi^2}
Out[4]= {e^{-2\pi^2}, e^{-8\pi^2}, e^{-18\pi^2}}
In[5]:= N[%]
Out[5]= {2.67529 \times 10^{-9}, 5.1225 \times 10^{-35}, 7.01996 \times 10^{-78}}
```

An excellent approximation is obtained by keeping just the first term in the sum, that is, $f(z) \approx \sqrt{2} \pi (1 + 2 e^{-2\pi^2} \cos(2\pi z))$.

■ *k*-ary Divisors

Q: The *k*-ary divisors of *n*, written $d \mid_k n$, are defined at mathworld.wolfram.com/k-aryDivisor.html. How can I implement $d \mid_k n$?

A: A divisor of a number *n* is a number *d* which divides *n* (written $d \mid n$), also called a factor. The divisors *d* of *n*, that is $d \mid n$, are computed using **Divisors**. Here are the divisors of 12.

```
In[1]:= Divisors[12]
Out[1]= {1, 2, 3, 4, 6, 12}
```

It is straightforward to implement the notation $d \mid n$.

```
In[2]:= << Utilities`Notation`
In[3]:= Notation[d_ | n_ => MemberQ[Divisors[n_], d_]]
```

Clearly 3 divides 12.

```
In[4]:= 3 | 12
Out[4]= True
```

A 1-ary or *unitary* divisor (mathworld.wolfram.com/UnitaryDivisor.html) *d* of *n* satisfies $d \perp n/d$, that is *d* is relatively prime to n/d or $\text{gcd}(d, n/d) = 1$. **divisors[1][*n*]** computes, and caches, the list of unitary divisors of *n*, $d \mid_1 n$.

```
In[5]:= divisors[1][n_Integer] :=
      divisors[1][n] = Select[Divisors[n], gcd[#, n/#] == 1 &]
```

Here are the unitary divisors of 12.

```
In[6]:= divisors[1][12]
Out[6]= {1, 3, 4, 12}
```

Define the notation $d \mid_k n$.

`In[7]:= Notation[d |k n \implies MemberQ[divisors[k][n], d]]`

In this notation, a divisor d of n is written $d |_0 n = d | n$.

`In[8]:= divisors[0][n_Integer] := Divisors[n]`

Here are the unitary divisors for the first few integers (Sloane's A077610).

`In[9]:= Table[{n, divisors[1][n]}, {n, 12}]`

`Out[9]=`

1	{1}
2	{1, 2}
3	{1, 3}
4	{1, 4}
5	{1, 5}
6	{1, 2, 3, 6}
7	{1, 7}
8	{1, 8}
9	{1, 9}
10	{1, 2, 5, 10}
11	{1, 11}
12	{1, 3, 4, 12}

For a prime power p^y , the unitary divisors are 1 and p^y [1].

`In[10]:= {#, divisors[1][#]} & /@ Prime[Range[10]]`

`Out[10]=`

2	{1, 2}
3	{1, 3}
5	{1, 5}
7	{1, 7}
11	{1, 11}
13	{1, 13}
17	{1, 17}
19	{1, 19}
23	{1, 23}
29	{1, 29}

2 is a divisor but not a unitary divisor of 12.

`In[11]:= {2 |0 12, 2 |1 12}`

`Out[11]= {True, False}`

d is called a k -ary divisor of n , written $d |_k n$, if the greatest common $(k-1)$ -ary divisor of d and n/d is 1. Recursive implementation, using caching, is straightforward.

`In[12]:= divisors[k_Integer][n_Integer] := divisors[k][n] =`

`Select[Divisors[n], Max[divisors[k-1][#] \cap divisors[k-1][$\frac{n}{\#}$]] == 1 &]`

Here are the 2-ary or *biunitary divisors* for the first few integers.

In[13]:= **Table[{n, divisors[2][n]}, {n, 12}]**

Out[13]=

1	{1}
2	{1, 2}
3	{1, 3}
4	{1, 4}
5	{1, 5}
6	{1, 2, 3, 6}
7	{1, 7}
8	{1, 2, 4, 8}
9	{1, 9}
10	{1, 2, 5, 10}
11	{1, 11}
12	{1, 3, 4, 12}

For a prime power p^y , the biunitary divisors are $1, p, p^2, \dots, p^y$ except for $p^{y/2}$ when y is even [1].

In[14]:= **Table[{n, 3^n |_2 3^6}, {n, 0, 6}]**

Out[14]=

0	True
1	True
2	True
3	False
4	True
5	True
6	True

Here are the k -ary divisors of 3^6 for $k = 0, 1, \dots, 10$. One observes that a fixed point is attained.

In[15]:= **Table[{k, divisors[k][3^6]}, {k, 0, 8}]**

Out[15]=

0	{1, 3, 9, 27, 81, 243, 729}
1	{1, 729}
2	{1, 3, 9, 81, 243, 729}
3	{1, 9, 81, 729}
4	{1, 3, 9, 81, 243, 729}
5	{1, 9, 81, 729}
6	{1, 9, 81, 729}
7	{1, 9, 81, 729}
8	{1, 9, 81, 729}

A divisor d of n is called *infinitary*, $d \mid_{\infty} n$, if it is a product of divisors of the form $p^{y_i 2^i}$, where p^y is a prime power dividing n and $\sum_i y_i 2^i$ is the binary representation of y (see Sloane's A037445). The infinitary divisors are then those factors d

that have zeros in the binary representation of all y_i s where n itself does. This can be directly coded using **BitOr**.

```
In[16]:= y[n_] := Select[Range[0, n], BitOr[n, #] == n &]
```

```
In[17]:= divisors[∞][n_] := Sort@Flatten@Outer[Times,
Sequence @@ (FactorInteger[n] /. {p_, m_Integer} :> py[m])]
```

```
In[18]:= divisors[∞][1] := {1}
```

p^x is an infinitary divisor of p^y (with $y > 0$) if $p^x \mid_{y-1} p^y$ [1].

```
In[19]:= 113 |6 117
```

```
Out[19]= True
```

```
In[20]:= 113 |∞ 117
```

```
Out[20]= True
```

Here are the infinitary divisors for the first few integers.

```
In[21]:= Table[{n, divisors[∞][n]}, {n, 16}]
```

```
Out[21]= {
  {1, {1}},
  {2, {1, 2}},
  {3, {1, 3}},
  {4, {1, 4}},
  {5, {1, 5}},
  {6, {1, 2, 3, 6}},
  {7, {1, 7}},
  {8, {1, 2, 4, 8}},
  {9, {1, 9}},
  {10, {1, 2, 5, 10}},
  {11, {1, 11}},
  {12, {1, 3, 4, 12}},
  {13, {1, 13}},
  {14, {1, 2, 7, 14}},
  {15, {1, 3, 5, 15}},
  {16, {1, 16}}
}
```

Taking the logarithm base p of the infinitary divisors of p^y gives the x values for which p^x are infinitary divisors of p^y .

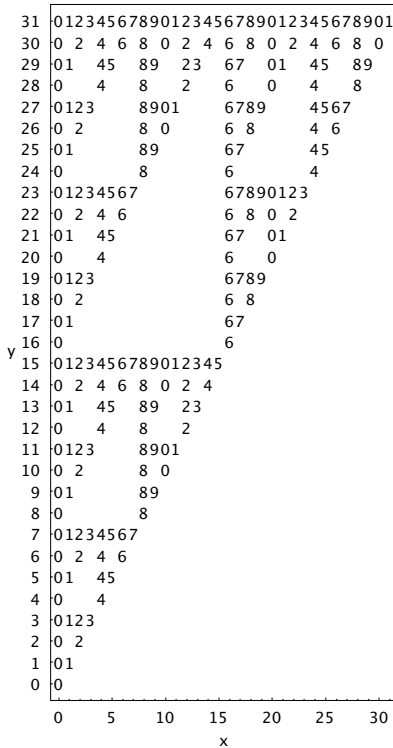
```
In[22]:= log2(divisors[∞])/@2Range[7]
```

```
Out[22]= {{0, 1}, {0, 2}, {0, 1, 2, 3}, {0, 4}, {0, 1, 4, 5}, {0, 2, 4, 6}, {0, 1, 2, 3, 4, 5, 6, 7}}
```

Here is a plot of the infinitary divisors of p^y , computed with $p = 2$ but valid for arbitrary $p \in \mathbb{P}$, for $y = 0, 1, \dots, 31$. Showing the last (decimal) digit of x enables the actual divisors to be read off easily [1].

```
In[23]:= Show[
  Graphics[MapIndexed[Text[Last[IntegerDigits[#1]], {#1, First[#2] - 1}] &,
    log2(divisors[∞]/@ 2Range[0,31]), {2}], Frame → True,
  FrameTicks → {Automatic, Range[0, 31], None, None}, AspectRatio → 2,
  TextStyle → {FontFamily → "MR", FontSize → 6}, FrameLabel → {x, y}]
```

From In[23]:=



The appearance of the Sierpinski sieve fractal mathworld.wolfram.com/SierpinskiSieve.html is interesting.

■ Asymptotic Expansion of a Sum

Q: Let r be an arbitrary irrational number. How can I find an asymptotic approximation to

$$f(x) = \sum_{n=1}^{\infty} \frac{x^n}{n-r} \quad (1)$$

as x approaches 1 from below?

A: The sum can be expressed as a Gaussian hypergeometric function.

$$\text{In}[1]:= f[r_][x_]=\sum_{n=1}^{\infty}\frac{x^n}{n-r}$$

$$\text{Out}[1]=-\frac{x {}_2F_1(1-r, 1; 2-r; x)}{r-1}$$

There are at least three advantages of this representation:

1. *Mathematica* includes arbitrary-precision algorithms for computing this function. For example, here is the sum with $r = \sqrt{541}$ and $x = 1 - 10^{-3}$.

$$\text{In}[2]:= N[f[\sqrt{541.}][1-10^{-3}], 20]$$

$$\text{Out}[2]= 0.21580785006326598375$$

2. Asymptotic approximations can be computed via series expansion. Introducing $x = 1 - \epsilon$, the following expansion is useful when $r \epsilon \ll 1$.

$$\text{In}[3]:= f[r][1-\epsilon]+O[\epsilon]^3 // \text{FullSimplify}$$

$$\text{Out}[3]= (-H_{-r} - \log(\epsilon)) + r(H_{-r} + \log(\epsilon) - 1)\epsilon - \frac{1}{4}((r-1)r(2H_{1-r} + 2\log(\epsilon) - 3))\epsilon^2 + O(\epsilon^3)$$

Here is the asymptotic approximation with $r = \sqrt{541}$ and $\epsilon = 10^{-3}$.

$$\text{In}[4]:= g[r_][\epsilon_] = \text{Normal}[\%];$$

$$\text{In}[5]:= g[\sqrt{541.}][10^{-3}]$$

$$\text{Out}[5]= 0.215812$$

3. The asymptotic series expansion in closed form can be obtained using functions.wolfram.com/07.23.06.0012.01,

$${}_2F_1(a, b; a+b; z) =$$

$$\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \sum_{k=0}^{\infty} \frac{(a)_k (b)_k}{k!^2} (2\psi(k+1) - \psi(a+k) - \psi(b+k) - \log(1-z))(1-z)^k /;$$

$$|1-z| < 1.$$

Here is the result.

$$\text{In}[6]:= f[r][1-\epsilon] /.$$

$${}_2F_1(a_ , b_ ; c_ ; z_) := \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \sum_{k=0}^{\infty} \frac{(a)_k (b)_k}{k!^2} (-\log(1-z) + 2\psi(k+1) - \psi(a+k) - \psi(b+k))(1-z)^k /; c == a+b // \text{FullSimplify}$$

$$\text{Out}[6]= -(\epsilon-1) \sum_{k=0}^{\infty} \frac{\epsilon^k (1)_k (1-r)_k (-\log(\epsilon) + \psi^{(0)}(k+1) - \psi^{(0)}(k-r+1))}{(k!)^2}$$

This can be used to compute the number of asymptotic terms required. For example, $k_{\max} = 5$ terms are required to get a result accurate to at least 10 decimal places with $r = \sqrt{541}$ and $\epsilon = 10^{-3}$.

```
In[7]:= With[{ $\epsilon = 10^{-3}$ ,  $r = \sqrt{541}$ .}],
Table[{ $k$ ,  $\frac{1}{(k!)^2} (\epsilon^k (1)_k (1-r)_k (-\log(\epsilon) + \psi^{(0)}(k+1) - \psi^{(0)}(k-r+1)))$ },
{ $k$ , 0, 5}]]
Out[7]=  $\begin{pmatrix} 0 & 0.24429 \\ 1 & -0.0286972 \\ 2 & 0.000434477 \\ 3 & -3.54558 \times 10^{-6} \\ 4 & 1.93943 \times 10^{-8} \\ 5 & -7.79834 \times 10^{-11} \end{pmatrix}$ 
```

■ Variable Order Simplify

Q: Suppose I have variables that are related in the following way: $d = a + b$ and $e = c + d$. Is there a straightforward way to get **FullSimplify** to consider these relationships, returning the simplest possible expression in terms of a, b, c, d , and e , when performing simplification?

A: Adam Strzebonski (adams@wolfram.com) answers: **VOISimplify**, a function for variable-order independent simplification that uses **FullSimplify** and considers all permutations of the variables, does this.

```
In[1]:= VOISimplify(vars_, expr_, assum_ : True) :=
Module[{perm, ee, best}, perm = Permutations[vars];
ee = (FullSimplify @@ ({expr, assum} /. Thread[vars -> #1]) &) /@ perm;
best = Sort[{LeafCount /@ ee, ee, perm}][[1]];
best[[2]] /. Thread[best[[3]] -> vars]]
```

Clearly, $d - b = a$.

```
In[2]:= VOISimplify({a, b, c, d, e}, d - b, {d = a + b, e = c + d})
```

```
Out[2]= a
```

Here is a more complicated example.

```
In[3]:= VOISimplify({a, b, c, d, e},
  a2 + 2 b a + 2 c a + b2 + c2 + d2 + e2 + 2 b c + 2 c e - 2 d e, {d = a + b, e = c + d})
Out[3]= (c + e)2
```

■ Simultaneous Recurrence Relations

Q: I have a set of recurrence relations, say

$$\begin{aligned} a_n &= 4 a_{n-1} (1 - a_{n-1}), \\ b_n &= 4 b_{n-1} (1 - b_{n-1}), \\ c_n &= \frac{1}{2} (a_n + b_n). \end{aligned}$$

How can I simplify them into a single recurrence relation involving only $c_n, c_{n-1}, c_{n-2}, \dots$?

A: Daniel Lichtblau (danl@wolfram.com) answers: To eliminate $a_n, a_{n-1}, b_n,$ and b_{n-1} , you need more equations. Write these equations as polynomials, including the polynomial for c_{n-1} .

```
In[1]:= polys[n_] = {a_n - 4 a_{n-1} (1 - a_{n-1}),
  b_n - 4 b_{n-1} (1 - b_{n-1}), c_n - 1/2 (a_n + b_n), c_{n-1} - 1/2 (a_{n-1} + b_{n-1})};
```

To acquire sufficiently many polynomials to eliminate all a_i and b_i variables, observe that putting $n \rightarrow n - 1$ gives a total of 7 polynomials in 9 variables.

```
In[2]:= allpolys = Union[polys[n], polys[n - 1]]
Out[2]= {a_{n-1} - 4 (1 - a_{n-2}) a_{n-2}, a_n - 4 (1 - a_{n-1}) a_{n-1},
  b_{n-1} - 4 (1 - b_{n-2}) b_{n-2}, b_n - 4 (1 - b_{n-1}) b_{n-1},
  1/2 (-a_{n-2} - b_{n-2}) + c_{n-2}, 1/2 (-a_{n-1} - b_{n-1}) + c_{n-1}, 1/2 (-a_n - b_n) + c_n}
```

```
In[3]:= Length[allpolys]
```

```
Out[3]= 7
```

```
In[4]:= vars = Variables[allpolys]
```

```
Out[4]= {a_{n-2}, a_{n-1}, a_n, b_{n-2}, b_{n-1}, b_n, c_{n-2}, c_{n-1}, c_n}
```

```
In[5]:= Length[vars]
```

```
Out[5]= 9
```

We want to eliminate 6 variables, **elims**, from the 7 polynomials.

```
In[6]:= cs = Cases[vars, c_]
```

```
Out[6]= {cn-2, cn-1, cn}
```

```
In[7]:= elims = Complement[vars, cs]
```

```
Out[7]= {an-2, an-1, an, bn-2, bn-1, bn}
```

Here is the desired relation.

```
In[8]:= Solve[Thread[allpolys == 0], cn, elims] // FullSimplify
```

```
Out[8]= {{cn → 64 (cn-2 - 1) cn-2 (1 - 2 cn-2)2 - 4 cn-12 + 4 (16 (cn-2 - 1) cn-2 + 5) cn-1}}
```

■ Aborting Messages

Q: I would like to abort a computation whenever an error occurs, avoiding evaluation of the subsequent expressions within a cell. For example, in the following computation, I would like to avoid computing $N[\zeta(2000)]$ after $\sin(2, 2)$ generates an error message.

```
In[1]:= sin(2, 2); N[ζ(2000)]
```

Sin::argx : Sin called with 2 arguments; 1 argument is expected. [More...](#)

```
Out[1]= 1.
```

One can immediately abort a computation by replacing the default for **\$Messages** with a call to **Abort**.

```
In[2]:= default = $Messages
```

```
Out[2]= {OutputStream[stdout, 1]}
```

```
In[3]:= $Messages := Abort[]
```

However, now when an error occurs, the computation is aborted without any message.

```
In[4]:= sin(2, 2); N[ζ(2000)]
```

```
Out[4]= $Aborted
```

Is there a way to display the message and then abort?

A: Omega Consulting (omegaconsultinggroup.com) answers: Using **\$Messages := Abort[]** is quite a creative trick. Unfortunately, you cannot use it and also get the message. **\$Messages** is a list of **OutputStreams** that any messages are written to. First, let us restore **\$Messages** to its default value.

```
In[5]:= $Messages = default;
```

If the **OutputStream** is **stdout**, then you get the normal message. If it is an **OutputStream** to a file, then it is written to that file. You might think that the following would work:

```
In[6]:= $Messages := {OutputStream["stdout", 1], Abort[]}
```

However, it does not.

```
In[7]:= sin(2, 2); N[ζ(2000)]
```

```
Out[7]= $Aborted
```

Why not? Because **\$Messages** is evaluated *before* the messages are actually sent to the streams. Again, we restore **\$Messages** to its default value.

```
In[8]:= $Messages = default;
```

Still there is another trick we can use, called a trap. A message is created whenever the **Message** function is called. We can redefine this function by unprotecting it.

```
In[9]:= Unprotect[Message];
```

Now we create a global variable, **\$AbortMessage**, to indicate what we want to do with a **Message**. When the variable is **True**, we want to deviate from the default behavior. When **False**, we want the default behavior (i.e., a message).

```
In[10]:= $AbortMessage = True;
```

Now we add a definition for when the variable is **True**. It calls **Message** with the variable set to **False** (creating a message) and then aborts.

```
In[11]:= Message[args___] :=
  (Block[{$AbortMessage = False}, Message[args]]; Abort[]) /; $AbortMessage
```

To be safe we reprotect **Message**.

```
In[12]:= Protect[Message];
```

Now we try it.

```
In[13]:= sin(2, 2); N[ζ(2000)]
```

Sin::argx : Sin called with 2 arguments; 1 argument is expected. [More...](#)

```
Out[13]= $Aborted
```

■ Notebook Options

Q : How do you add a new option to a **Notebook[]**?

A: John Fultz (jfultz@wolfram.com) answers: Use **SetOptions**. It is a good idea to store information using the **TaggingRules** option, which is specifically designated for this kind of use. For example,

```
In[1]:= SetOptions[InputNotebook[],
      TaggingRules → {NewOption → "new option string"}]
In[2]:= NewOption /. (TaggingRules /. Options[InputNotebook[], TaggingRules])
Out[2]= new option string
```

If you tried the **SetOptions** syntax directly on the option, it would work, but in a slightly unexpected way.

```
In[3]:= SetOptions[InputNotebook[], NewOption → "new option string"]
```

You will hear a beep. No result is returned for the usual syntax.

```
In[4]:= Options[InputNotebook[], NewOption]
```

Options::optnf : NewOption is not a known option for NotebookObject. [More...](#)

```
Out[4]= {}
```

However, the option is being saved.

```
In[5]:= NewOption /. Options[InputNotebook[]]
```

```
Out[5]= new option string
```

So, what is going on here? This is using a mechanism for forward compatibility. For example, what if a notebook from a future version with unknown options were to be read by an earlier version? It should just preserve options it does not understand. So, that is exactly what it is doing. **Options** on that specific option will not work because, internally, the option has been saved as a hidden option, but that hidden option always writes its contents out to files and the general **Options** requests.

■ References

- [1] G. L. Cohen, "On an Integer's Infinitary Divisors," *Mathematics of Computation*, **54**(189), 1990, pp. 395–411.

Paul Abbott

*School of Physics, M013
The University of Western Australia
35 Stirling Highway
Crawley WA 6009, Australia
tmj@physics.uwa.edu.au
physics.uwa.edu.au/~paul*