Ten Digit Problems

Lloyd N. Trefethen

Abstract. Most quantitative mathematical problems cannot be solved exactly, but there are powerful algorithms for solving many of them numerically to a specified degree of precision like ten digits or ten thousand. In this article three difficult problems of this kind are presented, and the story is told of the SIAM 100-Dollar, 100-Digit Challenge. The twists and turns along the way illustrate some of the flavor of algorithmic continuous mathematics.

1 Introduction

I am a mathematician who spends his time working with numbers, real numbers like 0.3233674316... and 22.11316746.... If I can compute a quantity to ten digits of accuracy, I am happy. Most mathematicians are not like this! In fact, sometimes it seems that the further you go in mathematics, the less important actual numbers become. But some of us develop algorithms to solve problems quantitatively, and we are called numerical analysts. I am the head of the Numerical Analysis Group at Oxford.

Like all mathematicians, I enjoy having a concrete problem to chew on. For example, what is the value of the integral

$$\int_0^1 x^{-1}\cos(x^{-1}\log x)\,dx\,? \qquad (1)$$

You won't find the answer in a table of integrals, and I don't think anybody knows how to derive an exact formula. But even though an exact formula does not exist, the integral still makes sense. (More precisely, it makes sense

Lloyd N. Trefethen
Oxford University Mathematical Institute, 24-29 St Giles, Oxford OX1 3LB, UK.
e-mail: `trefethen@maths.ox.ac.uk`

if we define (1) as the limit $\varepsilon \to 0$ of an integral from ε to 1.) The only way to evaluate it is by some kind of numerical algorithm, and this is hard, for the integrand (i.e., the function under the integral sign) oscillates infinitely often as x approaches 0 while swinging between larger and larger values that diverge to infinity. To ten digits, the answer turns out to be the first number listed above.

Each October in Oxford, four or five new graduate students arrive to begin a PhD in numerical analysis, and in their first term they participate in a course called the Problem Solving Squad. Each week I give them a problem like (1) whose solution is a single real number. Working in pairs, their job is to compute this number to as many digits of accuracy as they can. I don't give any hints, but the students are free to talk to anybody and use the library and the web. By the end of six weeks we always have some unexpected discoveries — and some tightly bonded graduate students!

In this article I want to tell you about three of these problems that have given me pleasure, which I'll call "two cubes", "five coins", and "blowup". Though this is the first article I've written about it, Oxford's Numerical Analysis Problem Solving Squad has been well known since the *SIAM 100-Dollar, 100-Digit Challenge* was organized in 2002. This involved ten problems selected from the early years of the Squad, and the challenge for contestants was to try to solve each problem to as many digits of accuracy as possible, up to ten digits for each. Teams from around the world entered the race, and twenty of them achieved perfect scores of 100 and won $100. Afterwards a book about the problems was published by four of the winners, with a cover picture illustrating an ingenious method of solving (1) using complex numbers [1]. I'll say more about the 100-Digit Challenge at the end.

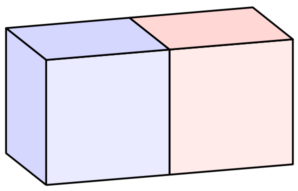
2 Two Cubes

Our first problem is motivated by a simple question from physics. Isaac Newton discovered that if two point masses of magnitude m_1 and m_2 are separated by a distance r, then they are attracted towards each other by a gravitational force of magnitude

$$F = \frac{Gm_1m_2}{r^2},$$

where G is a constant known as the *gravitational constant.* If you have masses that are not points but spheres or other objects, then each point in one mass is attracted to each point in the other by the same formula. We now pose the following idealized problem:

Problem 1. *Two objects of mass* 1 *attract each other gravitationally according to Newton's law with* $G = 1$. *Each object is a unit cube with its mass uniformly distributed. The centers of the cubes are one unit apart, so they are in contact along one face. What is the total force,* F*?*



You can think of the cubes as suns or planets if you like. No mathematician will be troubled by the idea of cubic planets. That's what mathematics is best at, reasoning about any kind of precisely defined situation, no matter how artificial. Something else about this problem, however, marks it out as unusual for most mathematicians. *It is so trivial!* We all know the formula for gravity, so where's the interest here? Working out the force between these particular bodies should be just a matter of bookkeeping. We are not in this business to be bookkeepers!

But some of us are in this business to design algorithms, and this innocent-looking problem is a killer. Let's try to solve it, and you'll see what I mean.

The first thought that may occur to you is, can't we replace each cube by a unit mass at the center and get the answer $F = 1$? Isn't that what Newton showed so many years ago, that as far as gravity is concerned, planets are equivalent to points? Well yes, Newton did show that, but only for spherical planets. If the shape is a cube, we have to investigate more carefully.

Let's say that cube 1 consists of points (x_1, y_1, z_1) with $0 < x_1, y_1, z_1 < 1$, and cube 2 consists of points (x_2, y_2, z_2) in the same range except $1 < x_2 < 2$. For unit point masses at (x_1, y_1, z_1) and (x_2, y_2, z_2), the force would be

$$\frac{1}{r^2} = \frac{1}{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}\,,$$

aligned in the direction between the points. Our job is to add up these forces over all pairs of points (x_1, y_1, z_1) and (x_2, y_2, z_2). That is, we need to evaluate a *six-dimensional integral.* The y and z components of the total force will cancel to zero, by symmetry, so it's the x component we need to integrate, which is equal to $(x_2 - x_1)/r$ times the expression above. That is, the x component of the force between unit masses at (x_1, y_1, z_1) and (x_2, y_2, z_2) is

$$f(x_1, y_1, z_1, x_2, y_2, z_2) = \frac{x_2 - x_1}{[(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2]^{3/2}}\,. \qquad (2)$$

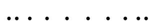
The number F we are looking for is thus

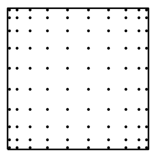
$$F = \int_0^1 \int_0^1 \int_1^2 \int_0^1 \int_0^1 \int_0^1 f(x_1, \ldots, z_2)\, dx_1\, dy_1\, dz_1\, dx_2\, dy_2\, dz_2\,. \qquad (3)$$

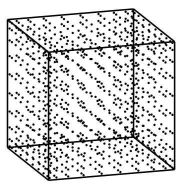
This is an integral over a six-dimensional cube. How do we turn it into a number?

I must be honest and confess that I am always behind schedule and usually make up these problems the night before giving them to the students. I aim to make sure I can compute a couple of digits at least, trusting that more powerful and beautiful ideas will come along during the week.

For this problem, the night before, I tried the most classical numerical method for evaluating integrals, *Gauss quadrature.* The idea of Gauss quadrature in one dimension is to sample the integrand at n precisely defined values of x called *nodes*, multiply the sampled values by n corresponding real numbers called *weights,* and add up the results. (The nodes and weights are determined by the condition that the estimate comes out exactly correct if the integrand happens to be a polynomial of degree no greater than $2n-1$.) For smooth integrands, such as those defined by functions that can be differentiated several times, this gives amazingly accurate approximations. And by squaring or cubing the grid, you can evaluate integrals in two or three dimensions. Here are pictures of 10, 10^2, and 10^3 Gauss nodes for integration over an interval, a square, and a cube:







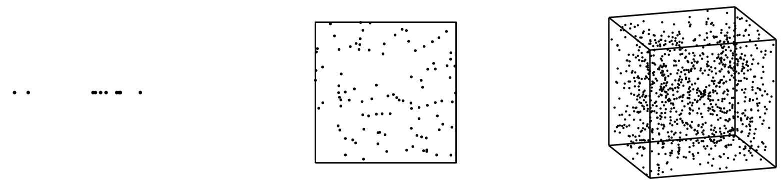
For our integral (3) the same idea applies, though it's not so easy to draw a picture.

Here is what I found with this method of "Gauss quadrature raised to the 6-th power". The number of nodes is $N = n^6$ with $n = 5, 10, \ldots, 30$, `FN` is the Gauss quadrature approximation to F, and `time` is the amount of time each computation took on my computer.

```
N =     15625   FN = 0.969313   time =   0.0 secs.
N =   1000000   FN = 0.947035   time =   0.3 secs.
N =  11390625   FN = 0.938151   time =   3.2 secs.
N =  64000000   FN = 0.933963   time =  17.6 secs.
N = 244140625   FN = 0.931656   time =  66.7 secs.
N = 729000000   FN = 0.930243   time = 198.2 secs.
```

This is awful! We can see that the answer looks like $F \approx 0.93$, 7% less than if the cubes were spheres. But that is all we can see, and it has taken minutes of computing time. Computing 10 digits would take pretty much forever.

In fact, these results from Gauss quadrature with its special nodes and weights are worse than what you get if you set all the weights equal to $1/N$ and place the nodes at random in the six-dimensional cube! This kind of randomized computation is called the *Monte Carlo method.* Here are typical sets of 10, 100, and 1000 random nodes in one, two and three dimensions:



And here's a set of Monte Carlo results for the same values of N as before.

```
N =     15625   FN = 0.906741   time =   0.1 secs.
N =   1000000   FN = 0.927395   time =   0.5 secs.
N =  11390625   FN = 0.925669   time =   4.4 secs.
N =  64000000   FN = 0.925902   time =  22.7 secs.
N = 244140625   FN = 0.926048   time =  88.0 secs.
N = 729000000   FN = 0.925892   time = 257.0 secs.
```

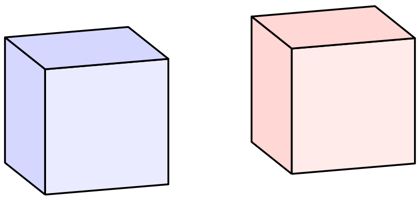
It seems that we now have three or four digits, $F \approx 0.9259$ or 0.9260. In this collection of results, and indeed for all the numbers reported in this article, it is a very interesting matter to try to make more precise statements about the accuracy of a computation. This is an important aspect of the field of numerical analysis, but to keep things as simple as possible, we shall settle for experimental evidence here and not attempt such estimates.

So, the world's slickest method for numerical integration is losing out to the world's simplest! Actually this often happens with high-dimensional integrals. The errors with Monte Carlo decrease in proportion to $1/\sqrt{N}$, the inverse of the square root of the number of samples, more or less independently of the number of dimensions, whereas Gauss quadrature slows down greatly with increasing dimension. This is a widespread theme in numerical algorithms, and one speaks of "the curse of dimensionality".

But even Monte Carlo hits a wall at 4 or 5 digits, or maybe 6 or 7 if we run overnight or use a parallel computer. How can we get more? The students worked hard and came up with many good ideas. Let's focus on one of these which eventually turned into a ten digit solution.

If you're familiar with Gauss quadrature, you can quickly spot why it has done so badly. The problem is that the integrand (2) is not smooth but *singular* because the cubes are right up against each other. The denominator goes to zero whenever $x_1 = x_2 = 1$, $y_2 = y_1$, and $z_2 = z_1$, so the fraction goes to ∞. This isn't bad enough to make the values of the integral infinite, but it slows down the convergence terribly.

We would like to eliminate the singularity. One way to do it would be to change the problem by separating the cubes, say, by a distance 1.



The convergence of Gauss quadrature changes completely, giving 14 digits in a fraction of a second:

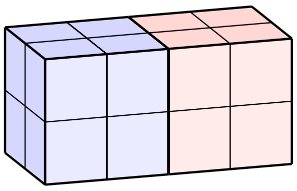
```
N =     15625   F = 0.24792296453612   time =  0.0 secs.
N =   1000000   F = 0.24792296916638   time =  0.3 secs.
N =  11390625   F = 0.24792296916638   time =  3.2 secs.
N =  64000000   F = 0.24792296916638   time = 17.6 secs.
```

Notice that the answer is close to 1/4, which is what the force would be if the cubes were spheres with centers separated by distance 2.

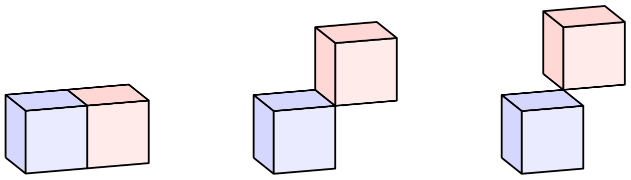
So we can accurately solve a modified problem, with the cubes separated. What about the original problem? Let $F(\varepsilon)$ denote the force between cubes separated by a distance $\varepsilon \geq 0$. We want to know $F(0)$, but we can only evaluate $F(\varepsilon)$ accurately for values of ε that are not too small. A good idea is to perform some kind of *extrapolation* from $\varepsilon > 0$ to $\varepsilon = 0$. Extrapolation is a well-developed topic in numerical mathematics, and some of the important methods in this area are known as Richardson extrapolation and Aitken extrapolation. The students and I tried a number of strategies like these and got... well, we were disappointed. We got another digit or two.

And then along came a delightful additional idea from graduate student Alex Prideaux, which finally nailed the two cubes problem.

Prideaux's idea was, let's break each cube into eight pieces, eight sub-cubes of size 1/2. Now the number F will be the sum of 64 pairwise contributions.



Four of these pairs meet along a face. Eight pairs meet along an edge, and four meet at a vertex:



In the other 48 cases the sub-cubes are well separated.

Having started with one kind of six-dimensional quadrature problem, we now have four! — face, edge, vertex, and separated. Let's define $\mathrm{Face}(d)$ to be the x component of the force between two cubes of size d touching along a face, and similarly $\mathrm{Edge}(d)$ and $\mathrm{Vertex}(d)$ for cubes of size d touching along edges and at vertices. If you think about the picture with 16 sub-cubes above, you will see that we can write equations for the forces at scale 1 in terms of the forces at scale $\frac{1}{2}$ like this:

$$\begin{aligned}\mathrm{Vertex}(1) &= \mathrm{Vertex}(\tfrac{1}{2}) + \text{well-separated terms},\\ \mathrm{Edge}(1) &= 2\,\mathrm{Edge}(\tfrac{1}{2}) + 2\,\mathrm{Vertex}(\tfrac{1}{2}) + \text{well-separated terms},\\ \mathrm{Face}(1) &= 4\,\mathrm{Face}(\tfrac{1}{2}) + 8\,\mathrm{Edge}(\tfrac{1}{2}) + 4\,\mathrm{Vertex}(\tfrac{1}{2}) + \text{well-separated terms}.\end{aligned}$$

This may look like dubious progress, until we note a basic fact of scaling:

$$\mathrm{Vertex}(\tfrac{1}{2}) = \tfrac{1}{16}\,\mathrm{Vertex}(1), \quad \mathrm{Edge}(\tfrac{1}{2}) = \tfrac{1}{16}\,\mathrm{Edge}(1), \quad \mathrm{Face}(\tfrac{1}{2}) = \tfrac{1}{16}\,\mathrm{Face}(1).$$

The factors of 16 come about as follows. If you halve the scale of a cubes problem, each mass decreases by a factor of 8, so the product of masses decreases by 64. On the other hand the distance between the cubes also cuts in half, so $1/r^2$ increases by a factor of 4. Thus overall, the force changes by the factor $4/64 = 1/16$.

Putting these observations together, we find

$$\begin{aligned}\mathrm{Vertex}(1) &= \tfrac{1}{16}\,\mathrm{Vertex}(1) + \text{well-separated terms},\\ \mathrm{Edge}(1) &= \tfrac{2}{16}\,\mathrm{Edge}(1) + \tfrac{2}{16}\,\mathrm{Vertex}(1) + \text{well-separated terms},\\ \mathrm{Face}(1) &= \tfrac{4}{16}\,\mathrm{Face}(1) + \tfrac{8}{16}\,\mathrm{Edge}(1) + \tfrac{4}{16}\,\mathrm{Vertex}(1) + \text{well-separated terms}.\end{aligned}$$

We can calculate the well-separated terms to high accuracy in a second or two by Gauss quadrature, and these formulas give us first $\mathrm{Vertex}(1)$, then $\mathrm{Edge}(1)$, and then the number we care about, $\mathrm{Face}(1)$. The answer is

$$F \approx 0.9259812606\,.$$

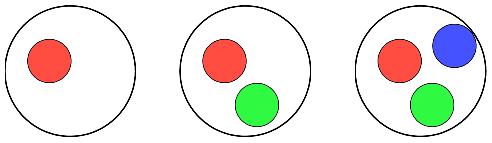
3 Five Coins

The second problem involves no physics, just geometry and probability.

Problem 2. *Non-overlapping coins of radius* 1 *are placed at random in a circle of radius* 3 *until no more can fit. What is the probability* p *that there are* 5 *coins?*

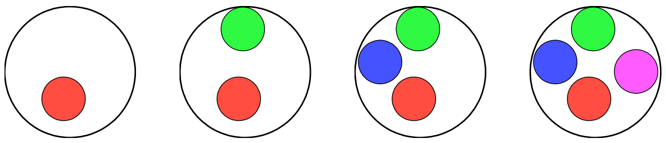
We shall see that this story, so far at least, has a less happy ending.

We can illustrate the game in a picture. We put a red coin down at random, then a green one, then a blue one. I'll leave it to you to prove that fitting in at least three coins is always possible. Here's an example that gets stuck at three, with no room for a fourth.

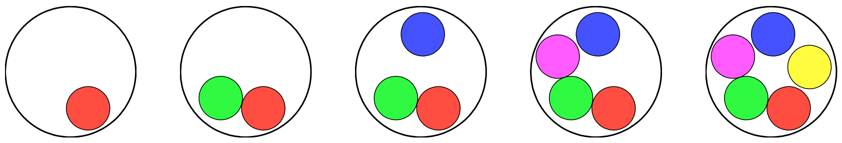


(Incidentally, to be precise the problem must specify what "at random" means. The meaning is pretty obvious; the trick is how to say it mathematically. If k coins are down, consider the set S of points at which the center of another coin could be placed. If S is nonempty, we put the center of coin $k+1$ at a point in S selected at random with respect to area measure.)

Quite often a fourth coin can fit too. Here's an example.



At four coins, we are usually finished. But occasionally a fifth can fit too:



Five coins is the limit. (Well, not quite. Six or seven are also possible, but the probabilities of these events are zero, meaning that no matter how many times you play the game randomly, you will never see them. Can you prove it? Think of where the centers of a 7-coin configuration would have to fall.)

So the question is, how often do we get to five coins? This problem has something in common with the two cubes. Since it is posed in terms of probability, one approach to a solution should be Monte Carlo simulation. We can write a computer program and see what happens. It's not obvious how best to organize the computation, but one reasonable approach is to replace the big disk by a fine grid, then pick points at random in that grid. Every time you pick a point, you remove it from further consideration along with all its neighbors at distance less than 2. For convergence to the required number, you must refine the grid and also increase the number of samples. By following this Monte Carlo approach we find approximately these frequencies:

3 coins: 18%   4 coins: 77%   5 coins: 5%

This level of accuracy can be achieved in 5 minutes. Running overnight gives possibly one more digit:

$$p \approx 0.053\,.$$

You may sense a bit of foreshadowing in the fact that we've printed this number in large type.

There's a scientific context to the five coins: this is known as a "parking problem". In one dimension, imagine a curb of length L with k cars parking at random, one after another, along the curb. How many cars will fit? Problems like this are of interest to chemists and physicists investigating aggregation of particles, and have been studied in 1, 2, and 3 dimensions. A question often asked is, in the limit of an infinite parking area, what fraction of the space can one expect will be filled by randomly arriving cars or coins or particles? In the one-dimensional case, the answer is known in the form of an integral that evaluates to $0.7475979202\ldots$.

For circular disks ("coins") in two dimensions, or spheres in three, we speak of a "Tanemura parking problem". So far as I am aware, no formula is known for the infinite-size limit in either of these situations.

But in any case our Problem 2 concerns not a limit but a very concrete setting of 3, 4, and 5 coins. And do you know something? Despite hard work, the Problem Squad never improved on 0.053. We found variations on the theme of Monte Carlo, but none that helped decisively. Yet this problem is one of finite-dimensional geometry, equivalent in fact to another multiple integral. There must be a way to solve it to high accuracy!

Sometimes a problem has no slick solution. In this case, I think a slick solution is still waiting to be found.

4 Blowup

Our final problem involves a partial differential equation (PDE). Since this may be unfamiliar territory, let me explain a little.

One of the best known PDEs is the *heat* or *diffusion equation:*

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}\,. \tag{4}$$

We have here a function $u(x, t)$ of a space variable x and a time variable t. The equation says that at each point in space-time, the partial derivative of u with respect to t is equal to its second partial derivative with respect to x. Physically, the idea is that at a particular point x and time t, the temperature will increase ($\partial u/\partial t > 0$) if the temperature curves upward as a function of x ($\partial^2 u/\partial x^2 > 0$), since this means that heat will flow in towards x from nearby hotter points. For example, one solution to (4) would be the function

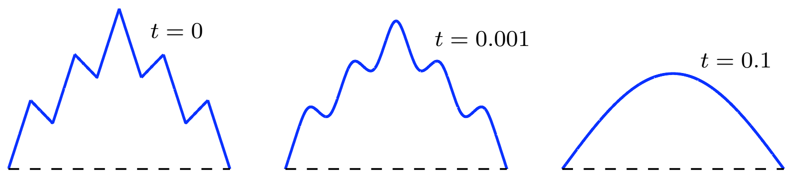
$$u(x,t) = e^{-t}\sin(x)\,,$$

for it is not hard to verify that for this choice of u,

$$\frac{\partial u}{\partial t} = -u\,, \qquad \frac{\partial^2 u}{\partial x^2} = -u\,.$$

In the time of Napoleon, Joseph Fourier discovered that equation (4) governs diffusion of heat in a one-dimensional body. For example, if a function $u_0(x)$ describes the temperature distribution in an infinitely long bar at time $t = 0$, then a solution $u(x,t)$ to (4) with initial condition $u(0,x) = u_0(x)$ tells the temperature at times $t > 0$. This was a first-class scientific discovery, and it is bad luck for Mr. Fourier that for whatever accidental reason of history, we talk about the Laplace and Poisson equations but not the Fourier equation.

Most PDE problems are posed on bounded domains, and then we must prescribe *boundary conditions* to determine the solution. For example, the heat equation might apply on the interval $x \in [-1,1]$, corresponding to a finite bar, with boundary conditions $u(-1,t) = u(1,t) = 0$, corresponding to zero temperature at both ends. Here's an illustration of a solution to this problem at different times. Notice how the sharp edges diffuse instantly, whereas the larger structure decays more slowly. That makes sense, since strong temperature differences between nearby points will quickly equalize.



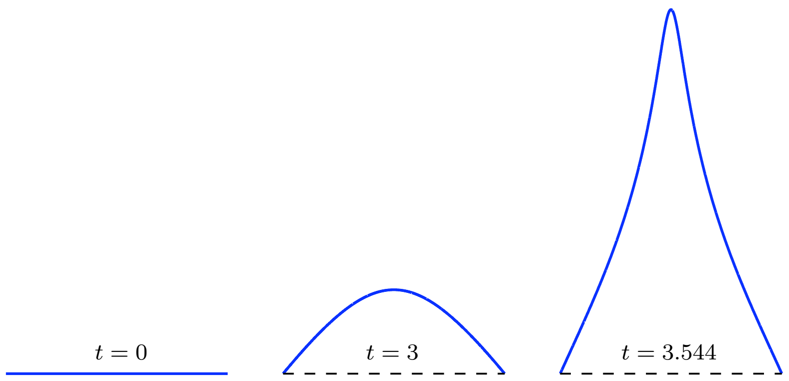
Eventually, all the heat flows out the ends and the signal decays to zero. (If you are troubled by how to take the second derivative of the jagged initial function, you are right to be troubled! We may imagine that $u(x,0)$ is a smooth function that happens to match the jagged curve to high accuracy.)

Equation (4) depends linearly on the variable u: it is a linear PDE. Our third problem involves a nonlinear PDE which consists of this equation plus an additional term, the exponential of u:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + e^u. \tag{5}$$

Whereas the heat equation just moves heat around, conserving the total heat apart from any inflow or outflow at the boundaries, this nonlinear term *adds* heat. You can think of e^u as a model of a chemical process like combustion, a temperature-dependent kind of combustion that accelerates exponentially as the temperature goes up.

Suppose we apply equation (5) on an interval $[-L, L]$ with initial condition $u(x,0) = 0$ and boundary conditions $u(-L,t) = u(L,t) = 0$. For $t > 0$, the exponential term adds heat, and the derivative term diffuses it out the boundaries. There's a competition here. If L is small, diffusion wins and the solution converges to a fixed limit function $u_\infty(x)$ as $t \to \infty$, with combustion exactly balancing diffusion. If L is larger, combustion wins. The heat can't diffuse away fast enough, and the solution explodes to infinity at a finite time $t = t_c$. In particular, this happens for the case $L = 1$. Here are the solutions at times 0, 3, and 3.544, by which point the amplitude has reached about 7.5. Soon afterwards, the curve will explode to infinity.



Physically, this blowup is related to the phenomenon of *spontaneous combustion.* Imagine a heap of grass cuttings or compost. Chemical processes may generate some heat, but if it's a small heap, the heat escapes and all is stable. In a larger pile, however, the heat may be unable to get away. Eventually, the heap catches fire. Much the same mathematics explains why a quantity of Uranium 235 has a *critical mass* above which it explodes in a fission chain reaction, the original principle behind atomic bombs.

Here then is our mathematical problem.

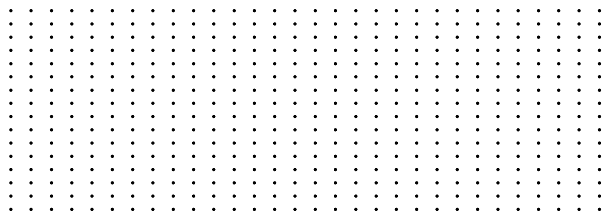
Problem 3. *At what time t_c does the solution $u(x,t)$ to the problem*

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + e^u, \qquad u(x,0) = 0, \quad u(-1,t) = u(1,t) = 0 \tag{6}$$

blow up to infinity?

The numerical solution of PDEs on computers goes back to von Neumann and others in the 1940s. It is as important a topic as any in numerical analysis, and a huge amount is known. For Problem 3, the geometry is an interval and the equation is a simple one with a single variable. Other problems of interest to scientists and engineers may be much more complicated. The shapes of wings and airplanes are designed by solving PDEs of fluid and structural mechanics in complicated three-dimensional geometries. Weather forecasts come from solutions of PDE problems involving variables representing air velocity, temperature, pressure, humidity and more, and now the geometry is nothing less than a chunk of planet Earth with its oceans and islands and mountains.

Most numerical solutions of PDEs depend on discretizing the problem, replacing partial derivatives by finite approximations. The grid around an airplane may be eye-poppingly complicated, but for (6), one might begin by using a simple regular grid like this one, with the horizontal direction corresponding to x and the vertical direction to t.



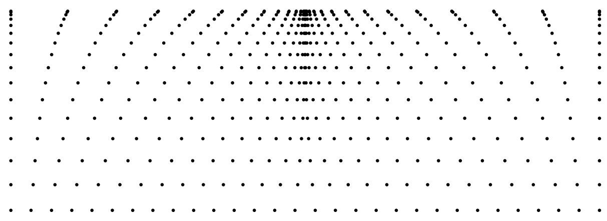
Quite a good solution strategy for Problem 3 is to discretize (6) on grids like this, shrink the step sizes Δx and Δt systematically, and then use some kind of extrapolation to estimate the blowup time.

For example, one way to discretize this equation from $t = 0$ to $t = 3.544$ is to divide $[-1, 1]$ into N space intervals and $[0, 3.544]$ into $2N^2$ time intervals and then approximate the PDE on this grid in a manner whose details we won't go into. Here are the approximate values $u(0, 3.544)$ produced by this method for a succession of values of N:

```
N =   32   u(0,3.544) = 9.1015726   time =   0.0 secs.
N =   64   u(0,3.544) = 7.8233770   time =   0.1 secs.
N =  128   u(0,3.544) = 7.5487013   time =   0.6 secs.
N =  256   u(0,3.544) = 7.4823971   time =   3.3 secs.
N =  512   u(0,3.544) = 7.4659568   time =  21.2 secs.
N = 1024   u(0,3.544) = 7.4618549   time = 136.2 secs.
```

It seems clear that the true value $u(0, 3.544)$ must be about 7.46, and by applying Richardson extrapolation to the data, one can improve this estimate to 7.460488. Using methods like this, with some ingenuity and care, one can estimate the blow-up time for Problem 3 to six or seven digits.

It seems wasteful, however, to use a regular grid for a problem like this where all the action happens in a narrow spike near $x = 0$ and $t = 3.5$. It is tempting to try to take advantage of this structure by using some kind of uneven grid, one which gets finer as the spike gets narrower, like this:



Adapting grids like this to the solution being computed is a big topic in numerical PDE. When flows are computed over airplanes, the grids may be thousands of times finer near the surface than further out.

The ten digit solution that I know of for Problem 3 makes use of a highly nontrivial adaptive-gridding algorithm due to my former student Wynn Tee. Tee's method starts from the observation that although (6) is posed for x in the interval $[-1, 1]$, we can extend the solution to complex values of x too, that is, to values of x with an imaginary as well as a real part. As t approaches t_c, it turns out that the solution $u(x, t)$ has singularities in the complex x-plane that approach the real axis. By monitoring this situation and using what is known as a conformal map to distort the grid systematically, one can maintain extremely high accuracy with a small number of grid points even as the spike grows very tall and narrow. In fact it is possible to calculate a solution to ten digits of accuracy with only 100 grid points in the x direction:

$$t_c \approx 3.544664598\,.$$

This solution also takes advantage of advanced methods of time discretization, and it is really a tour de force of clever computation, illustrating that some very abstract mathematics may be useful for concrete problems.

5 The 100-Digit Challenge

What does it mean to solve a mathematical problem? That's too big a question, for the solution to a mathematical problem might be a "yes" or "no", a proof, a counterexample, a theorem, who knows what. More specifically, then: what does it mean to solve one of those mathematical problems whose solution, in principle, is a number? Must we find an exact formula — and if we do, is that good enough regardless of the formula's complexity? Must we write down the number in decimal form — and how many digits are enough? Is it sufficient to find an algorithm that can generate the number — and does it matter how quickly it runs?

There are plenty of sand traps in this discussion. Even the notion of an exact formula is elusive. For example, in the theory of roots of polynomials, an exact formula is traditionally allowed to include n-th roots, like $\sqrt[3]{2}$, but not trigonometric or other special functions, like $\sin(2)$. A computer, however, doesn't see much difference between $\sqrt[3]{2}$ and $\sin(2)$. Both are calculated by fast iterative algorithms. For that matter, even a fraction like 2/3 is calculated by an iterative algorithm on some computers, but nobody would question the credentials of 2/3 as an exact solution! What about a more complicated expression like $\int_0^1 e^{-x^4} dx$? That's easy for your computer too, but it will probably have to call a piece of software rather than use something built-in to the microprocessor hardware. Is it an exact solution?

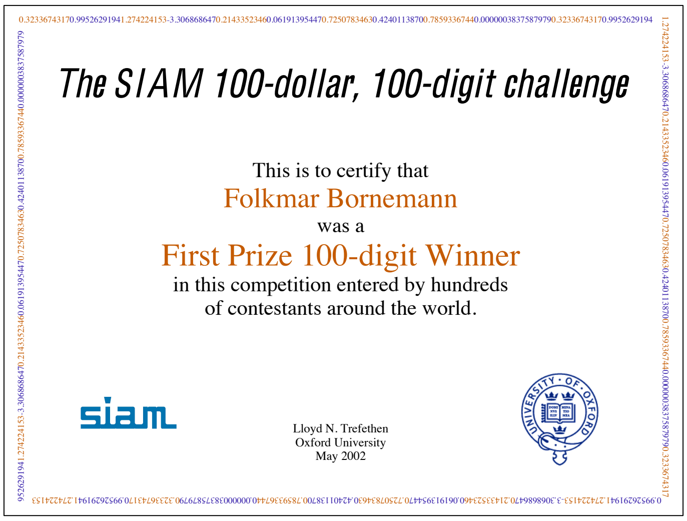
For me, solving a numerical problem means finding an algorithm that can calculate the answer to high accuracy on a computer, whether or not there's an explicit formula, and this brings us to the question: what's so special about ten digits? Why not three digits, or a hundred? I'd like to suggest two reasons why ten digits is indeed a good goal to aim for.

One reason is that in science, many things are known to more than three digits of accuracy, but hardly anything is known to more than ten. If a quantity is known to a hundred or a million digits, you can be sure it is a mathematical abstraction like π rather than a physical constant like the speed of light or Planck's constant. So in science, you might say that ten digits is more or less the same as infinity. Ultimately this is why computers normally compute with 16-digit precision, not 160. (And since 10 digits is comfortably less than 16, you usually don't have to worry too much about computer precision when tackling a ten-digit problem.)

The second reason is illustrated by the five coins. To exaggerate a little bit, you can solve just about any problem to three digits of accuracy by brute force. But a brute force algorithm doesn't encode much insight, and it often fails if you try to push it much further. This is just what happened with the five coins, where we got stuck at three digits with Monte Carlo. Ten digits is a very different achievement. To get to ten digits, you need a good understanding of your problem and a well targeted algorithm. In fact, if you can get this far, the chances are pretty good that you could get ten thousand if you had to. To see what I mean, let's return to the 100-Digit Challenge.

The Challenge was launched in January 2002, and its ten problems involved the integral (1), some chaotic dynamics, the norm of an infinite matrix, global optimization in two dimensions, the approximation of the gamma function in the complex plane, a random walk on a lattice, inverting a $20\,000 \times 20\,000$ matrix, the heat equation on a square plate, parametrized optimization, and Brownian motion. Each team was allowed to have up to six members, and 94 teams entered from 25 countries. Twenty of them got perfect scores! That surprised me. I had planned to spend \$100 rewarding whoever got the most digits, but with twenty perfect scores, I was unsure what to do. To my great pleasure, a donor came forward to plug the gap — William Browning, founder of Applied Mathematics, Inc. in Connecticut. You might think that for a member of a team of six to give nights and weekends to a mathematical project and be rewarded with \$16.67 must be some kind of sour joke. But it turned out that receiving a little bit of cash meant a great deal to the winners as a symbolic recognition of their achievement.

The winners also got certificates, like this one awarded to Folkmar Bornemann of the Technical University of Munich, one of the authors of the book *The SIAM 100-Digit Challenge* [1].



0.32336743170.99526291941.274224153-3.30686864 70.21433523460.06191395447 0.72507834630.42401138700.78593367440.00000383758797 90.32336743170.9952629194

The SIAM 100-dollar, 100-digit challenge

This is to certify that

Folkmar Bornemann

was a

First Prize 100-digit Winner

in this competition entered by hundreds of contestants around the world.

siam

Lloyd N. Trefethen
Oxford University
May 2002

UNIVERSITY · OF · OXFORD

I published an article in *SIAM News* telling the story and outlining a solution to each problem. The article ended like this:

> If you add together our heroic numbers, the result is $\tau = 1.497258836\ldots$. I wonder if anyone will ever compute the ten thousandth digit of this fundamental constant?

Now I wrote this to be funny, and to get people thinking. What's funny is that this number τ is the most unfundamental constant you could imagine. The sum of the answers to ten arbitrary unrelated problems — what nonsense! I chose the Greek letter tau because privately I was thinking of this as "Trefethen's Constant". With good British modesty I felt it was OK to name something after oneself, provided the item was sufficiently ridiculous.

In the book [1], τ took on a life of its own. The authors amazed us all by finding ways to solve nine of the problems, one after another, to ten thousand digits! The variety of mathematical, algorithmic, and computational tools they used was striking. Indeed, the book emphasizes throughout that there is no "right way" to solve a problem, and your bag of tools can never be too big. By one beautiful chain of reasoning making use of ideas of the Indian mathematician Ramanujan, Bornemann found an exact formula for the solution to Challenge Problem 10 (Brownian motion). By another remarkable method based on ideas related to the field of number theory, Jean-Guillaume Dumas together with 186 computer processors running for four days were able to compute exactly the solution to Challenge Problem 7 (inverting a matrix): the task was to find a particular element of this inverse, and they discovered that the answer was a rational number equal to the quotient of two 97,389-digit integers.

Bornemann et al. wrote an appendix called "Extreme Digit-Hunting", which reported their super-accurate results in this format:

```
0.32336 74316 77778 76139 93700 <<9950 digits>> 42382 81998 70848 26513 96587 27
```

The listing goes to digit 10,002, so that if you added up the ten numbers, you'd be confident that the 10,000-th digit of the sum would be correct.

But Challenge Problem 3 proved intractable. (This was to determine what is called the "2-norm" of a matrix with infinitely many rows and columns, with entries $a_{11} = 1$, $a_{12} = 1/2$, $a_{21} = 1/3$, $a_{13} = 1/4$, $a_{22} = 1/5$, $a_{31} = 1/6, \ldots$.) With a month of computer time the authors computed 273 digits:

```
1.2742 24152 82122 81882 12340 <<220 digits>> 75880 55894 38735 33138 75269 029
```

And that's where Trefethen's constant is stuck as of today, at 273 digits.

I decided at age 20 to devote my career to number crunching, and it has given me unending satisfaction since then. My knowledge and confidence have advanced with the years, and so have the computers and software tools available. What a feeling, to be working on algorithms related to those that control spacecraft, design integrated circuits, and run satellite navigation systems — yet still be so close to elegant mathematics!

The field of numerical analysis can be defined like this:

Numerical analysis is the study of algorithms
for the problems of continuous mathematics.

This means problems involving real or complex variables, not just integers. "Continuous" is the opposite of "discrete", and algorithms for discrete problems have quite a different flavor and a different community of experts. Like any scientific field, numerical analysis stretches over a pure–applied range, with some people spending most of their time inventing algorithms or applying them to scientific problems, while others are more interested in rigorous analysis of their properties. In earlier centuries the leading pure mathematicians were the same people as the leading numerical ones, like Newton and Euler and Gauss, but mathematics has grown a lot since then, and now the two groups are rather separate. If you look at numbers of specialists, numerical analysis is nowadays one of the biggest branches of mathematics.

Let's finish with another ten digit problem from my file. Suppose you have three identical regular tetrahedra, each of volume 1. What's the volume of the smallest sphere you can fit them inside?

Every problem is different. This is the only one so far for which I've found myself playing with cardboard models! I made three tetrahedra, then juggled and jiggled till I thought I knew roughly what the shape of the optimal configuration must be. By numerically minimizing a function whose derivation took hours of tricky trigonometry, I got the estimate

$$22.113167462973\ldots.$$

Now by a curious coincidence, my computer tells me

$$256\pi(\sqrt{12}-\sqrt{10})^3 = 22.113167462973\ldots.$$

Have we stumbled upon the exact answer? I think so, but I'm not sure, and I certainly don't have a proof. And what in the world gave me the idea of calculating $256\pi(\sqrt{12}-\sqrt{10})^3$?[1]

6 Epilogue

Stop Press! We've had an unexpected development on Problem 1, the two cubes. I showed a draft of this article to Prof. Bengt Fornberg of the University of Colorado, one of the best numerical problem-solvers in the world. He got hooked. The problem is so simple, yet so devilishly hard!

Working with pencil and paper and the symbolic computing system Mathematica, Fornberg managed to shrink the dimensionality from six to five, then four. Then three, then two. That is, he managed to reduce Problem 1 to a two-dimensional integral to be evaluated numerically. As he peeled off one dimension after another, the formulas kept getting more complicated, and he fought hard to keep the complexity under control.

Then one morning Fornberg reported he was down to one dimension. This meant that the problem could be five-sixths solved analytically, leaving just a one-dimensional integral to be evaluated numerically. We were startled.

The next morning, Fornberg had the exact solution! It was preposterously long and complicated. He kept working, making more and more simplifications of trigonometric functions and logarithms and hyperbolic functions and their inverses, combining some terms together and also splitting some terms into two to make the result more elementary. And here is what he found:

$$\begin{aligned} F = \frac{1}{3}\biggl(&\frac{26\pi}{3} - 14 + 2\sqrt{2} - 4\sqrt{3} + 10\sqrt{5} - 2\sqrt{6} + 26\log(2) - \log(25) \\ &+ 10\log(1+\sqrt{2}) + 20\log(1+\sqrt{3}) - 35\log(1+\sqrt{5}) \\ &+ 6\log(1+\sqrt{6}) - 2\log(4+\sqrt{6}) - 22\tan^{-1}(2\sqrt{6})\biggr)\ . \end{aligned}$$

[1] OK, I'll tell you. My calculation led to the estimate $R \approx 0.85368706$ for the radius of the smallest sphere that can enclose three tetrahedra each of side length 1. I tried typing this number into the Inverse Symbolic Calculator at `http://oldweb.cecm.sfu.ca/projects/ISC/ISCmain.html`, and back came the suggestion that R might be $4(\sqrt{6}-\sqrt{5})$. Cubing this number and multiplying by $4\pi/3$ gives the volume of the sphere, and dividing that result by $\sqrt{2}/12$, which is the volume of a regular tetrahedron with side length 1, gives $256\pi(\sqrt{12}-\sqrt{10})^3$.

So now we can have as many digits as we want:

$$F \approx 0.925981260557291428093436687 0\ldots.$$

Most computational problems don't have exact solutions, but when I cook up challenges for the Problem Squad, the drive for elegance keeps me close to the edge of tractability. In this case we were lucky.

Further Reading

[1] Folkmar Bornemann, Dirk Laurie, Stan Wagon, and Jörg Waldvogel, *The SIAM* 100*-digit challenge: a study in high-accuracy numerical computing.* SIAM, Philadelphia, 2004.

[2] Jonathan M. Borwein and David H. Bailey, *Mathematics by experiment: plausible reasoning in the 21st century.* A K Peters, Natick/MA, 2003.

[3] Timothy Gowers, June Barrow-Green, and Imre Leader (editors), *The Princeton companion to mathematics.* Princeton University Press, Princeton/NJ, 2008.

[4] T. Wynn Tee and Lloyd N. Trefethen, *A rational spectral collocation method with adaptively transformed Chebyshev grid points.* SIAM Journal of Scientific Computing **28** (2006), 1798–1811.

[5] Lloyd N. Trefethen, *Ten digit algorithms.* Numerical Analysis Technical Report NA-05/13, Oxford University Computing Laboratory; `www.comlab.ox.ac.uk/oucl/publications/natr/`.