MATH 100 – Introduction to the Profession

The Need for Approximation

Greg Fasshauer

Department of Applied Mathematics
Illinois Institute of Technology

Fall 2014

Outline[1]

1. Why Bother With Approximations?
2. Pure Math
3. Computational Math
4. Theoretical Computer Science
5. Mathematical Modeling

[1] Part of this comes from [T. Gowers: Mathematics: A Very Short Introduction, Chapter 7], while some other topics may relate back to phenomena seen earlier.

"Real" Math Isn't Always "Clean"

Most people think of mathematics as a very clean, exact subject. One learns at school to expect . . . a simple formula. Those who continue with mathematics at university level . . . soon discover that nothing could be further from the truth. For many problems it would be miraculous and totally unexpected if somebody were to find a precise formula for the solution; most of the time one must settle for a rough estimate instead. Until one is used to estimates, they seem ugly and unsatisfying. However, it is worth acquiring a taste for them, because not to do so is to miss out on many of the greatest theorems and most interesting unsolved problems in mathematics.

from [T. Gowers: Mathematics: A Very Short Introduction]

Knowledge of a (rough) estimate or an approximate answer is important in many areas of math:

- Pure math: many results in number theory are proved using models and estimates based on probabilistic arguments (see, e.g., the prime number theorem below).
- Computational math: absolutely crucial in order to know how accurate, how reliable and how fast numerical algorithms are (see, e.g., Cramer's rule and derivative approximation below).
- Theoretical computer science: similarly important to estimating the run-time of algorithms (see, e.g., the famous ***P*** vs. ***NP*** problem below).
- Math modeling: often so-called asymptotic analysis is used.

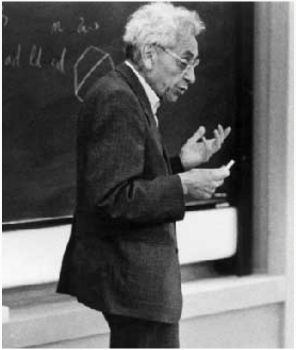
Prime Numbers

Since Euclid we know that there are infinitely many primes (see lecture on proofs), but. . .

Prime Numbers

Since Euclid we know that there are infinitely many primes (see lecture on proofs), but. . .

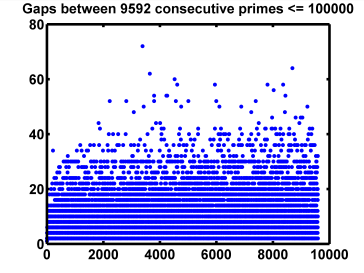
> *God may not play dice with the universe, but something strange is going on with the prime numbers.*
>
> Paul Erdős



Prime Numbers

Since Euclid we know that there are infinitely many primes (see lecture on proofs), but. . .

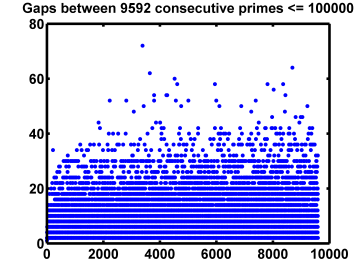
> *God may not play dice with the universe, but something strange is going on with the prime numbers.*
>
> Paul Erdős



Prime Numbers

Since Euclid we know that there are infinitely many primes (see lecture on proofs), but. . .

> *God may not play dice with the universe, but something strange is going on with the prime numbers.*
>
> Paul Erdős



Problem: We don't really understand prime numbers and their distribution – and yet they have vast practical applications, especially in security algorithms (cryptography, RSA encoding, . . .).

Prime Numbers

Since Euclid we know that there are infinitely many primes (see lecture on proofs), but. . .

> *God may not play dice with the universe, but something strange is going on with the prime numbers.*
>
> Paul Erdős

Problem: We don't really understand prime numbers and their distribution – and yet they have vast practical applications, especially in security algorithms (cryptography, RSA encoding, . . .).

Possibly **the** biggest unsolved math problem: the Riemann hypothesis.

Approximate Information for Primes

Gauss investigated the distribution of prime numbers.
He defined $\pi(N)$, the number of primes up to N, and conjectured

$$\pi(N) \approx \frac{1}{\ln 2} + \frac{1}{\ln 3} + \frac{1}{\ln 4} + \ldots + \frac{1}{\ln N}$$

based on his observation that the density of prime numbers is related to logarithms, i.e.,

$$\text{density} = \frac{\text{mass}}{\text{volume}} = \frac{\pi(N)}{N} = \frac{\text{number of primes}}{\text{length of interval}} \approx \frac{1}{\ln N} = \frac{1}{\text{average gap}}.$$

Approximate Information for Primes

Gauss investigated the distribution of prime numbers.
He defined $\pi(N)$, the number of primes up to N, and conjectured

$$\pi(N) \approx \frac{1}{\ln 2} + \frac{1}{\ln 3} + \frac{1}{\ln 4} + \ldots + \frac{1}{\ln N}$$

based on his observation that the density of prime numbers is related to logarithms, i.e.,

$$\text{density} = \frac{\text{mass}}{\text{volume}} = \frac{\pi(N)}{N} = \frac{\text{number of primes}}{\text{length of interval}} \approx \frac{1}{\ln N} = \frac{1}{\text{average gap}}.$$

See the MATLAB function `PrimeTheorem(N)` for this and a few other examples,

Approximate Information for Primes

Gauss investigated the distribution of prime numbers.
He defined $\pi(N)$, the number of primes up to N, and conjectured

$$\pi(N) \approx \frac{1}{\ln 2} + \frac{1}{\ln 3} + \frac{1}{\ln 4} + \ldots + \frac{1}{\ln N}$$

based on his observation that the density of prime numbers is related to logarithms, i.e.,

$$\text{density} = \frac{\text{mass}}{\text{volume}} = \frac{\pi(N)}{N} = \frac{\text{number of primes}}{\text{length of interval}} \approx \frac{1}{\ln N} = \frac{1}{\text{average gap}}.$$

See the MATLAB function `PrimeTheorem(N)` for this and a few other examples, and Marcus Du Sautoy's music of the primes for more on the Riemann hypothesis.

Twin Primes Conjecture & Bounded Gaps

Even though the prime number theorem tells us that average gaps between primes are increasing, we also have

Twin primes: a pair of primes separated by a gap of 2, e.g.,

$$(3,5),(5,7),(11,13),(17,19),\dots$$

Twin Primes Conjecture & Bounded Gaps

Even though the prime number theorem tells us that average gaps between primes are increasing, we also have

Twin primes: a pair of primes separated by a gap of 2, e.g.,

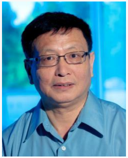
$$(3,5),(5,7),(11,13),(17,19),\ldots$$

It's conjectured (see the earlier plot – but not proved) that there are infinitely many such pairs.

Twin Primes Conjecture & Bounded Gaps

Even though the prime number theorem tells us that average gaps between primes are increasing, we also have

Twin primes: a pair of primes separated by a gap of 2, e.g.,



$$(3,5), (5,7), (11,13), (17,19), \ldots$$

It's conjectured (see the earlier plot – but not proved) that there are infinitely many such pairs.

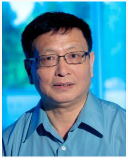
In 2013, Yitang Zhang proved that there are infinitely many pairs of primes that differ by at most 70,000,000.

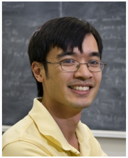
Twin Primes Conjecture & Bounded Gaps

Even though the prime number theorem tells us that average gaps between primes are increasing, we also have

Twin primes: a pair of primes separated by a gap of 2, e.g.,

$$(3,5),(5,7),(11,13),(17,19),\ldots$$





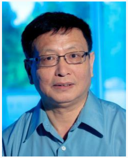
It's conjectured (see the earlier plot – but not proved) that there are infinitely many such pairs.
In 2013, Yitang Zhang proved that there are infinitely many pairs of primes that differ by at most 70,000,000.
Since then – as a polymath project started by Terence Tao – this number was reduced to 4680.

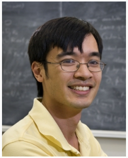
Twin Primes Conjecture & Bounded Gaps

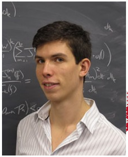
Even though the prime number theorem tells us that average gaps between primes are increasing, we also have

Twin primes: a pair of primes separated by a gap of 2, e.g.,

$$(3,5),(5,7),(11,13),(17,19),\ldots$$







It's conjectured (see the earlier plot – but not proved) that there are infinitely many such pairs.

In 2013, Yitang Zhang proved that there are infinitely many pairs of primes that differ by at most 70,000,000.

Since then – as a polymath project started by Terence Tao – this number was reduced to 4680. Next James Maynard improved it to 600, and then – using Maynard's ideas – the polymath project brought it to 246 (which is still pretty far from 2, but much closer than 70,000,000 – or no bound at all!).

Approximating Derivatives

By dropping the limit from the definition of the derivative we can approximate the value $y'(t)$ of some function y by using a forward difference approximation

$$y'(t) \approx \frac{y(t+h) - y(t)}{h}.$$

Approximating Derivatives

By dropping the limit from the definition of the derivative we can approximate the value $y'(t)$ of some function y by using a forward difference approximation

$$y'(t) \approx \frac{y(t+h) - y(t)}{h}.$$

While we can show that this approximation is exact for linear functions, you may be suspicious of this approach.

Approximating Derivatives

By dropping the limit from the definition of the derivative we can approximate the value $y'(t)$ of some function y by using a forward difference approximation

$$y'(t) \approx \frac{y(t+h) - y(t)}{h}.$$

While we can show that this approximation is exact for linear functions, you may be suspicious of this approach.

So, how accurate is it for more general functions?

Using a Taylor series expansion (see MATH 152) we can show that

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2} y''(\tau),$$

where τ is somewhere between t and $t+h$.

Using a Taylor series expansion (see MATH 152) we can show that

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2}y''(\tau),$$

where τ is somewhere between t and $t+h$.
This yields

$$y'(t) = \frac{y(t+h) - y(t)}{h} - \frac{h}{2}y''(\tau),$$

so that the truncation error of the forward difference approximation depends on

Using a Taylor series expansion (see MATH 152) we can show that

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2}y''(\tau),$$

where τ is somewhere between t and $t+h$.
This yields

$$y'(t) = \frac{y(t+h) - y(t)}{h} - \frac{h}{2}y''(\tau),$$

so that the truncation error of the forward difference approximation depends on

- the step size h, and
- the magnitude of the second derivative y''.

Using a Taylor series expansion (see MATH 152) we can show that

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2}y''(\tau),$$

where τ is somewhere between t and $t+h$.
This yields

$$y'(t) = \frac{y(t+h) - y(t)}{h} - \frac{h}{2}y''(\tau),$$

so that the truncation error of the forward difference approximation depends on

- the step size h, and
- the magnitude of the second derivative y''.

See how this works using `DerivativeApproximation.m`.

Using a Taylor series expansion (see MATH 152) we can show that

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2}y''(\tau),$$

where τ is somewhere between t and $t+h$.
This yields

$$y'(t) = \frac{y(t+h) - y(t)}{h} - \frac{h}{2}y''(\tau),$$

so that the truncation error of the forward difference approximation depends on

- the step size h, and
- the magnitude of the second derivative y''.

See how this works using `DerivativeApproximation.m`.
Clearly, we get better and better approximations by simply making h smaller[2].

[2]Other derivative approximation methods give "more bang for the buck" by having a truncation error that goes to zero like h^2 as $h \to 0$ (see MATH 350).

How do we estimate the cost of Cramer's rule?

Cramer's rule states that the solution $\boldsymbol{x} = (x_1, \ldots, x_n)^T$ of the linear system $\mathsf{A}\boldsymbol{x} = \boldsymbol{b}$ is given by

$$x_i = \frac{\det \mathsf{A}_i}{\det \mathsf{A}}, \qquad i = 1, \ldots, n,$$

where A_i is obtained from A by replacing its i^{th} column by $\boldsymbol{b}$. Therefore, we need to compute $n+1$ determinants, which can be shown to require approximately $3n!$ arithmetic operations each[3].

[3] Using the standard recursive approach.

How do we estimate the cost of Cramer's rule?

Cramer's rule states that the solution $\boldsymbol{x} = (x_1, \ldots, x_n)^T$ of the linear system $\mathsf{A}\boldsymbol{x} = \boldsymbol{b}$ is given by

$$x_i = \frac{\det \mathsf{A}_i}{\det \mathsf{A}}, \qquad i = 1, \ldots, n,$$

where A_i is obtained from A by replacing its i^{th} column by $\boldsymbol{b}$. Therefore, we need to compute $n+1$ determinants, which can be shown to require approximately $3n!$ arithmetic operations each[3].

	Flops				
n	10^9 (Giga)	10^{10}	10^{11}	10^{12} (Tera)	10^{15} (Peta)
10	10^{-1}sec	10^{-2}sec	10^{-3}sec	10^{-4}sec	negligible
15	17 hours	1.74 hours	10.46 min	1 min	$0.6\ 10^{-1}$ sec
20	4860 years	486 years	48.6 years	4.86 years	1.7 day
25	o.r.	o.r.	o.r.	o.r.	38365 years

[3] Using the standard recursive approach.

The *P* vs. *NP* Problem

One task of theoretical computer science is to determine the complexity of algorithms.

As we saw with Cramer's rule (or with the recursive Fibonacci algorithm earlier), it is often not feasible to run an algorithm with very large input size, and so runtime must be estimated.

The *P* vs. *NP* Problem

One task of theoretical computer science is to determine the complexity of algorithms.
As we saw with Cramer's rule (or with the recursive Fibonacci algorithm earlier), it is often not feasible to run an algorithm with very large input size, and so runtime must be estimated.
Another Millennium problem asks

> *whether questions exist whose answer can be quickly checked, but which require an impossibly long time to solve by any direct procedure.*

(see `http://www.claymath.org/millenium-problems/p-vs-np-problem`)

The ***P*** vs. ***NP*** Problem

One task of theoretical computer science is to determine the complexity of algorithms.

As we saw with Cramer's rule (or with the recursive Fibonacci algorithm earlier), it is often not feasible to run an algorithm with very large input size, and so runtime must be estimated.

Another Millennium problem asks

> *whether questions exist whose answer can be quickly checked, but which require an impossibly long time to solve by any direct procedure.*

(see `http://www.claymath.org/millenium-problems/p-vs-np-problem`)

In other words, is the class ***P*** (of "easily solvable" problems) the same as the class ***NP*** (of "easily checkable" problems)?

The ***P*** vs. ***NP*** Problem

One task of theoretical computer science is to determine the complexity of algorithms.
As we saw with Cramer's rule (or with the recursive Fibonacci algorithm earlier), it is often not feasible to run an algorithm with very large input size, and so runtime must be estimated.
Another Millennium problem asks

> *whether questions exist whose answer can be quickly checked, but which require an impossibly long time to solve by any direct procedure.*

(see `http://www.claymath.org/millenium-problems/p-vs-np-problem`)
In other words, is the class ***P*** (of "easily solvable" problems) the same as the class ***NP*** (of "easily checkable" problems)?
See Ian Stewart's excellent analysis of the game Minesweeper.

Asymptotic or Perturbation Analysis

In modeling situations, one strategy is to write a problem in dimensionless form, and then analyze what happens for a parameter (known to be either small or large).

Asymptotic or Perturbation Analysis

In modeling situations, one strategy is to write a problem in dimensionless form, and then analyze what happens for a parameter (known to be either small or large).

Example (from [S. Howison: Practical Applied Mathematics])

Consider the quadratic equation (with small ε)

$$\varepsilon x^2 + x - 1 = 0$$

and find a solution without using the quadratic formula.

Asymptotic or Perturbation Analysis

In modeling situations, one strategy is to write a problem in dimensionless form, and then analyze what happens for a parameter (known to be either small or large).

Example (from [S. Howison: Practical Applied Mathematics])

Consider the quadratic equation (with small ε)

$$\varepsilon x^2 + x - 1 = 0$$

and find a solution without using the quadratic formula.
Idea: Consider the desired solution as a perturbation of the much simpler problem

$$x - 1 = 0,$$

which arises for $\varepsilon = 0$.

To get an approximate solution for the original (perturbed) problem we make an Ansatz

$$x = 1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots$$

To get an approximate solution for the original (perturbed) problem we make an Ansatz

$$x = 1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots$$

Now substitute this into the original equation:

$$\varepsilon(1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots)^2 + (1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots) - 1 = 0$$

To get an approximate solution for the original (perturbed) problem we make an Ansatz

$$x = 1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots$$

Now substitute this into the original equation:

$$\varepsilon(1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots)^2 + (1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots) - 1 = 0$$

Reordering according to powers of ε we get

$$\varepsilon(1 + x_1) + \varepsilon^2(x_2 + 2x_1) + \ldots = 0,$$

To get an approximate solution for the original (perturbed) problem we make an Ansatz

$$x = 1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots$$

Now substitute this into the original equation:

$$\varepsilon(1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots)^2 + (1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots) - 1 = 0$$

Reordering according to powers of ε we get

$$\varepsilon(1 + x_1) + \varepsilon^2(x_2 + 2x_1) + \ldots = 0,$$

and so (comparing coefficients)

$$x_1 = -1, \qquad x_2 = -2x_1 = 2.$$

To get an approximate solution for the original (perturbed) problem we make an Ansatz

$$x = 1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots$$

Now substitute this into the original equation:

$$\varepsilon(1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots)^2 + (1 + \varepsilon x_1 + \varepsilon^2 x_2 + \ldots) - 1 = 0$$

Reordering according to powers of ε we get

$$\varepsilon(1 + x_1) + \varepsilon^2(x_2 + 2x_1) + \ldots = 0,$$

and so (comparing coefficients)

$$x_1 = -1, \qquad x_2 = -2x_1 = 2.$$

This gives (from the *Ansatz*)

$$x = 1 - \varepsilon + 2\varepsilon^2 + \ldots$$

as approximate solution to the original quadratic equation.

"Real" use of asymptotic analysis is called for in applications such as

- orbit calculations in astronomy,
- stability analysis of differential equations,
- boundary layers of differential equations as arise in fluid flow problems (e.g., water waves),
- modeling of lubricants,
- and many others.

See MATH 486 for more.

References I

- Gowers, Timothy.
  Mathematics: A Very Short Introduction.
  Oxford University Press, 2002.
- Howison, Sam.
  Practical Applied Mathematics: Modelling, Analysis, Approximation.
  Cambridge University Press, 2005.
- Sautoy, Marcus Du.
  The Music of the Primes: Searching to Solve the Greatest Mystery in Mathematics.
  Harper Collins, 2004.
- C. Moler.
  Experiments with MATLAB.
  Free download at
  http://www.mathworks.com/moler/exm/chapters.html

References II

A. Quarteroni, F. Saleri and P. Gervasio.
Scientific Computing with MATLAB and Octave (3rd ed.).
Springer, Berlin, 2010.