Math 317: Operations Research
First Lecture

Steven J Miller
Williams College

sjm1@williams.edu

http://www.williams.edu/Mathematics/sjmiller/public_html/317

Williams College

Introduction and Objectives

Introduction / Objectives

Main Topic: Optimization: Linear Programming.

Objectives

- Obviously learn linear programming.
- Emphasize techniques / asking the right questions.
- Model problems and analyze model.
- Elegant solutions vs brute force.
- Writing textbook for AMS.

Types of Problems

- Diet problem.
- Banking (asset allocation).
- Scheduling (movies, airlines, TSP, MLB).
- Elimination numbers.
- Sphere packing....

My (applied) experiences

- Marketing: parameters for linear programming (SilverScreener).
- Data integrity: detecting fraud with Benford's Law (IRS, Iranian elections).
- Sabermetrics: Pythagorean Won-Loss Theorem.

Course Mechanics

Grading / Administrative

- HW: 15%. Midterm 40%. Final/Project 40%. Class Participation 5%. May change a bit. A large portion of work/grade from a group project: you'll give a talk, prepare a well-crafted manuscript, and respectfully listen to reports of others.

- Pre-reqs: linear algebra (analysis, stats, programming a plus).

Office hours / feedback

- TBD and when I'm in my office (schedule online).
- Feedback ephsmath@gmail.com, password first 8 Fibonacci numbers (011235813).

Other

- Webpage: numerous handouts, additional comments each day (mix of review and optional advanced material).
- Opportunity to help write a book.
- PREPARE FOR CLASS! Must do readings before each class.

Other: Advice from Jeff Miller

- Party less than the person next to you.

Other: Advice from Jeff Miller

- Party less than the person next to you.
- Take advantage of office hours / mentoring.

Other: Advice from Jeff Miller

- Party less than the person next to you.
- Take advantage of office hours / mentoring.
- Learn to manage your time: no one else wants to.

Other: Advice from Jeff Miller

- Party less than the person next to you.
- Take advantage of office hours / mentoring.
- Learn to manage your time: no one else wants to.

Happy to do practice interviews, adjust deadlines....

Linear algebra textbooks online: `http://joshua.smcvt.edu/linalg.html/book.pdf`

Useful links

LaTeX and Mathematica Tutorials and Templates

http://web.williams.edu/Mathematics/sjmiller/public_html/math/handouts/latex.htm

Has templates for using LaTeX for papers, talks, posters, and a Mathematica tutorial.

Also videos on each.

Examples / Jobs

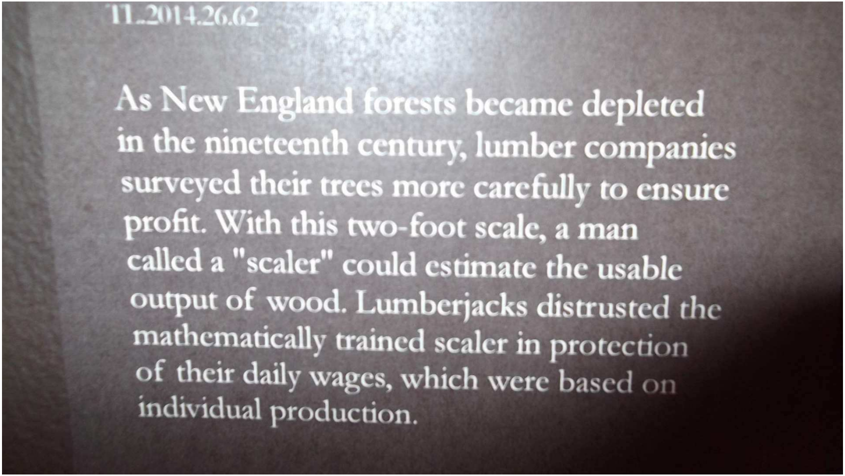
Alabama vs Auburn: 2013

https://www.youtube.com/watch?v=sLO2SmM9gPw

Log ruler (and WCMA)

Log ruler (and WCMA)

Log ruler (and WCMA)



Scheduling: Baseball Tournaments, Swim Lessons

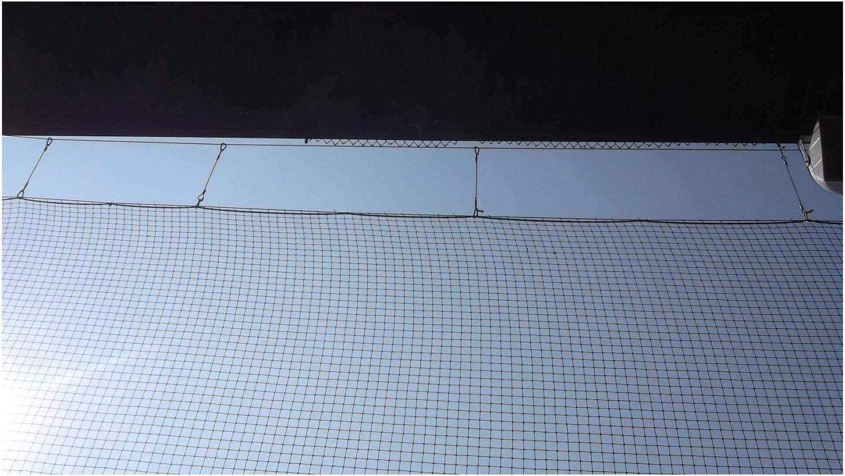
Scheduling: Baseball Tournaments, Swim Lessons

Scheduling: Baseball Tournaments, Swim Lessons

Inefficiencies from Location

Inefficiencies from Location

Inefficiencies from Location



Inefficiencies from Location

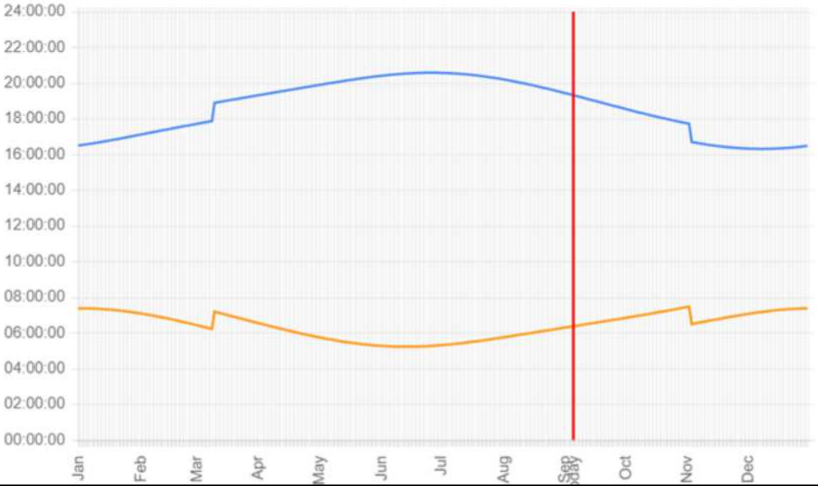
Inefficiencies from Location

Inefficiencies from Location

Maps

Year distribution of sunrise and sunset times in North Adams, MA – 2019

https : // sunrise – sunset.org / us / north – adams – ma



Maps

Who America is rooting for in the Super Bowl:

Pascal's Triangle

Pascal's Triangle

Video on Pascal's Triangle

https://www.youtube.com/watch?v=tt4_4YajqRM

Fast Multiplication

Cost of Standard Polynomial Evaluation

Multiplication far more expensive than addition....

$f(x) = 3x^5 - 8x^4 + 7x^3 + 6x^2 - 9x + 2$: Cost is $5 + 4 + 3 + 2 + 1 + 0 = 15$ multiplications.

These are triangle numbers: degree d have $d(d+1)/2$.

Cost of Standard Polynomial Evaluation

Multiplication far more expensive than addition....

$f(x) = 3x^5 - 8x^4 + 7x^3 + 6x^2 - 9x + 2$: Cost is $5 + 4 + 3 + 2 + 1 + 0 = 15$ multiplications.

These are triangle numbers: degree d have $d(d+1)/2$.

$$\begin{aligned} S(d) &= 1 + 2 + \cdots + d \\ S(d) &= d + (d-1) + \cdots 1 \end{aligned}$$

Cost of Standard Polynomial Evaluation

Multiplication far more expensive than addition....

$f(x) = 3x^5 - 8x^4 + 7x^3 + 6x^2 - 9x + 2$: Cost is $5 + 4 + 3 + 2 + 1 + 0 = 15$ multiplications.

These are triangle numbers: degree d have $d(d+1)/2$.

$$\begin{aligned} S(d) &= 1 + 2 + \cdots + d \\ S(d) &= d + (d-1) + \cdots 1 \end{aligned}$$

Thus $2S(d) = d \cdot (d+1)$ and claim follows.

Horner's Algorithm

$f(x) = 3x^5 - 8x^4 + 7x^3 + 6x^2 - 9x + 2$: Cost is $5 + 4 + 3 + 2 + 1 + 0 = 15$ multiplications.

Horner's algorithm:

$$\Bigg(\bigg(\Big(\big(3x - 8\big)x + 7\Big)x + 6\bigg)x - 9\Bigg)x + 2.$$

Horner's Algorithm

$f(x) = 3x^5 - 8x^4 + 7x^3 + 6x^2 - 9x + 2$: Cost is $5 + 4 + 3 + 2 + 1 + 0 = 15$ multiplications.

Horner's algorithm:

$$\Bigg(\bigg(\Big(\big(3x - 8\big)x + 7\Big)x + 6\bigg)x - 9\Bigg)x + 2.$$

Cost is degree d multiplications!

Useful also in fractal plotting.... Shows can often do common tasks faster.

Fast Multiplication

Horner is best in general, but maybe for special polynomials can do better?

Try polynomials of the form $f(x) =$

Fast Multiplication

Horner is best in general, but maybe for special polynomials can do better?

Try polynomials of the form $f(x) = x^n$.

Write n in binary: Say $n = 100 = 64 + 32 + 4 = 1100100_2$.

Fast Multiplication

Horner is best in general, but maybe for special polynomials can do better?

Try polynomials of the form $f(x) = x^n$.

Write n in binary: Say $n = 100 = 64 + 32 + 4 = 1100100_2$.

$$
\begin{aligned}
x \cdot x &= x^2 \\
x^2 \cdot x^2 &= x^4 \\
x^4 \cdot x^4 &= x^8 \\
x^8 \cdot x^8 &= x^{16} \\
x^{16} \cdot x^{16} &= x^{32} \\
x^{32} \cdot x^{32} &= x^{64}
\end{aligned}
$$

Fast Multiplication

Horner is best in general, but maybe for special polynomials can do better?

Try polynomials of the form $f(x) = x^n$.

Write n in binary: Say $n = 100 = 64 + 32 + 4 = 1100100_2$.

$$
\begin{aligned}
x \cdot x &= x^2 \\
x^2 \cdot x^2 &= x^4 \\
x^4 \cdot x^4 &= x^8 \\
x^8 \cdot x^8 &= x^{16} \\
x^{16} \cdot x^{16} &= x^{32} \\
x^{32} \cdot x^{32} &= x^{64}
\end{aligned}
$$

Fast Multiplication

Horner is best in general, but maybe for special polynomials can do better?

Try polynomials of the form $f(x) = x^n$.

Write n in binary: Say $n = 100 = 64 + 32 + 4 = 1100100_2$.

$$\begin{aligned} x \cdot x &= x^2 \\ x^2 \cdot x^2 &= x^4 \\ x^4 \cdot x^4 &= x^8 \\ x^8 \cdot x^8 &= x^{16} \\ x^{16} \cdot x^{16} &= x^{32} \\ x^{32} \cdot x^{32} &= x^{64} \end{aligned}$$

Fast Multiplication

Horner is best in general, but maybe for special polynomials can do better?

Try polynomials of the form $f(x) = x^n$.

Write n in binary: Say $n = 100 = 64 + 32 + 4 = 1100100_2$.

$$\begin{aligned} x \cdot x &= x^2 \\ x^2 \cdot x^2 &= x^4 \\ x^4 \cdot x^4 &= x^8 \\ x^8 \cdot x^8 &= x^{16} \\ x^{16} \cdot x^{16} &= x^{32} \\ x^{32} \cdot x^{32} &= x^{64} \end{aligned}$$

Recap

Horner takes us from order d^2 to order d.

Fast multiplication takes us to order $\log_2 d$, but only for special polynomials; these though are the ones used in RSA!

Euclidean Algorithm

Preliminaries

Input x, y with $y > x$.

Goals: find $\gcd(x, y)$, find a, b so that $ax + by = \gcd(x, y)$.

Lot of ways to go: non-constructive proofs of a, b but need values; Euclidean algorithm is *very* fast.

Euclidean Algorithm

Let $r_0 = y, r_1 = x$.

$$r_0 = q_1 r_1 + r_2, \quad 0 \leq r_2 < r_1.$$

Euclidean Algorithm

Let $r_0 = y, r_1 = x$.

$$r_0 = q_1 r_1 + r_2, \quad 0 \leq r_2 < r_1.$$

$$r_1 = q_2 r_2 + r_3, \quad 0 \leq r_3 < r_2.$$

Euclidean Algorithm

Let $r_0 = y, r_1 = x$.

$$r_0 = q_1 r_1 + r_2, \quad 0 \leq r_2 < r_1.$$

$$r_1 = q_2 r_2 + r_3, \quad 0 \leq r_3 < r_2.$$

Continue until....

$$r_n = q_{n+1} r_{n+1} + r_{n+2}, \quad r_{n+2} \in \{0, 1\}.$$

Euclidean Algorithm

Let $r_0 = y, r_1 = x$.

$$r_0 = q_1 r_1 + r_2, \quad 0 \le r_2 < r_1.$$

$$r_1 = q_2 r_2 + r_3, \quad 0 \le r_3 < r_2.$$

Continue until....

$$r_n = q_{n+1} r_{n+1} + r_{n+2}, \quad r_{n+2} \in \{0, 1\}.$$

Note $\gcd(r_0, r_1) = \gcd(r_1, r_2) = \gcd(r_2, r_3), \ldots$.

Euclidean Algorithm

Let $r_0 = y, r_1 = x$.

$$r_0 = q_1 r_1 + r_2, \quad 0 \leq r_2 < r_1.$$

$$r_1 = q_2 r_2 + r_3, \quad 0 \leq r_3 < r_2.$$

Continue until....

$$r_n = q_{n+1} r_{n+1} + r_{n+2}, \quad r_{n+2} \in \{0, 1\}.$$

Note $\gcd(r_0, r_1) = \gcd(r_1, r_2) = \gcd(r_2, r_3), \ldots$.

Can 'climb upwards' to get a, b such that $ax + by = \gcd(x, y)$.

Fermat's little Theorem

Euler totient function

$\phi(n)$ is the number of integers from 1 to n relatively prime to n.

$\phi(p) = p - 1$ and $\phi(pq) = (p-1)(q-1)$ if p, q distinct primes.

Do not need, but $\phi(mn) = \phi(m)\phi(n)$ if $\gcd(m, n) = 1$, and $\phi(p^k) = p^k - p^{k-1}$.

A lot of group theory lurking in the background, only doing what absolutely need.

Fermat's little Theorem

Fermat's little Theorem (FlT)

Let a be relatively prime to n. Then $a^{\phi(n)} = 1 \mod n$.

Special cases: $a^{p-1} = 1 \mod p$, $a^{(p-1)(q-1)} = 1 \mod pq$.

Will only prove these two cases....

Proof of Fermat's little Theorem: $n = p$

Proof: Let $n = p$, let $\gcd(a, p) = 1$.

Consider $1, 2, \ldots, p-1$ and $a, 2a, \ldots, (p-1)a$.

Claim both sets are all residues modulo p.

Proof of Fermat's little Theorem: $n = p$

Proof: Let $n = p$, let $\gcd(a, p) = 1$.

Consider $1, 2, \ldots, p-1$ and $a, 2a, \ldots, (p-1)a$.

Claim both sets are all residues modulo p.

If $ia = ja \bmod p$ then $(i-j)a = 0 \bmod p$ so $i = j \bmod p$.

Proof of Fermat's little Theorem: $n = p$

Proof: Let $n = p$, let $\gcd(a, p) = 1$.

Consider $1, 2, \ldots, p - 1$ and $a, 2a, \ldots, (p - 1)a$.

Claim both sets are all residues modulo p.

If $ia = ja \bmod p$ then $(i - j)a = 0 \bmod p$ so $i = j \bmod p$.
Thus $(p - 1)! = (p - 1)!a^{p-1} \bmod p$, so $a^{p-1} = 1 \bmod p$. $\square$

Proof of Fermat's little Theorem: $n = p$

Proof: Let $n = p$, let $\gcd(a, p) = 1$.

Consider $1, 2, \ldots, p-1$ and $a, 2a, \ldots, (p-1)a$.

Claim both sets are all residues modulo p.

If $ia = ja \bmod p$ then $(i-j)a = 0 \bmod p$ so $i = j \bmod p$.
Thus $(p-1)! = (p-1)!a^{p-1} \bmod p$, so $a^{p-1} = 1 \bmod p$. $\square$

Note: General case: $x_1, \ldots, x_{\phi(n)}$ and $ax_1, \ldots, ax_{\phi(n)}$.

Proof of Fermat's little Theorem: $n = pq$

Proof: Let $n = pq$, let $\gcd(a, pq) = 1$.

Proof of Fermat's little Theorem: $n = pq$

Proof: Let $n = pq$, let $\gcd(a, pq) = 1$.

Apply FIT with a^{q-1} and p: $(a^{q-1})^{p-1} = 1 \bmod p$.

Apply FIT with a^{p-1} and q: $(a^{p-1})^{q-1} = 1 \bmod q$.

Proof of Fermat's little Theorem: $n = pq$

Proof: Let $n = pq$, let $\gcd(a, pq) = 1$.

Apply FIT with a^{q-1} and p: $(a^{q-1})^{p-1} = 1 \bmod p$.

Apply FIT with a^{p-1} and q: $(a^{p-1})^{q-1} = 1 \bmod q$.

Thus $a^{(p-1)(q-1)}$ is $1 \mod p$ and is $1 \bmod q$.

$a^{(p-1)(q-1)} = 1 + \alpha p = 1 + \beta q$.

Proof of Fermat's little Theorem: $n = pq$

Proof: Let $n = pq$, let $\gcd(a, pq) = 1$.

Apply FIT with a^{q-1} and p: $(a^{q-1})^{p-1} = 1 \mod p$.

Apply FIT with a^{p-1} and q: $(a^{p-1})^{q-1} = 1 \mod q$.

Thus $a^{(p-1)(q-1)}$ is $1 \mod p$ and is $1 \mod q$.

$a^{(p-1)(q-1)} = 1 + \alpha p = 1 + \beta q$.

Thus $\alpha p = \beta q$ so $q|\alpha$ and $p|\beta$, so $a^{(p-1)(q-1)} = 1 \mod pq$.
$\square$

Primality Tests from FIT

If $\gcd(a, n) = 1$ and $a^{n-1} \neq 1 \mod n$ then n cannot be prime.

If equalled 1 then n ***might be*** prime.

Primality Tests from FIT

If $\gcd(a, n) = 1$ and $a^{n-1} \neq 1 \mod n$ then n cannot be prime.

If equalled 1 then n ***might be*** prime.

- If can take high powers, very fast!
- Can suggest candidate primes, and then use better, slower test for certainty.
- Carmichael numbers: Composites that are never rejected: 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, 15841, 29341, ... (OEIS A002997).