Experimental Evaluation of the Height of a Random Set of Points in a d-dimensional Cube

Eric Breimer
Mark Goldberg
Brian Kolstad
Malik Magdon-Ismail
Rensselaer Polytechnic Institute

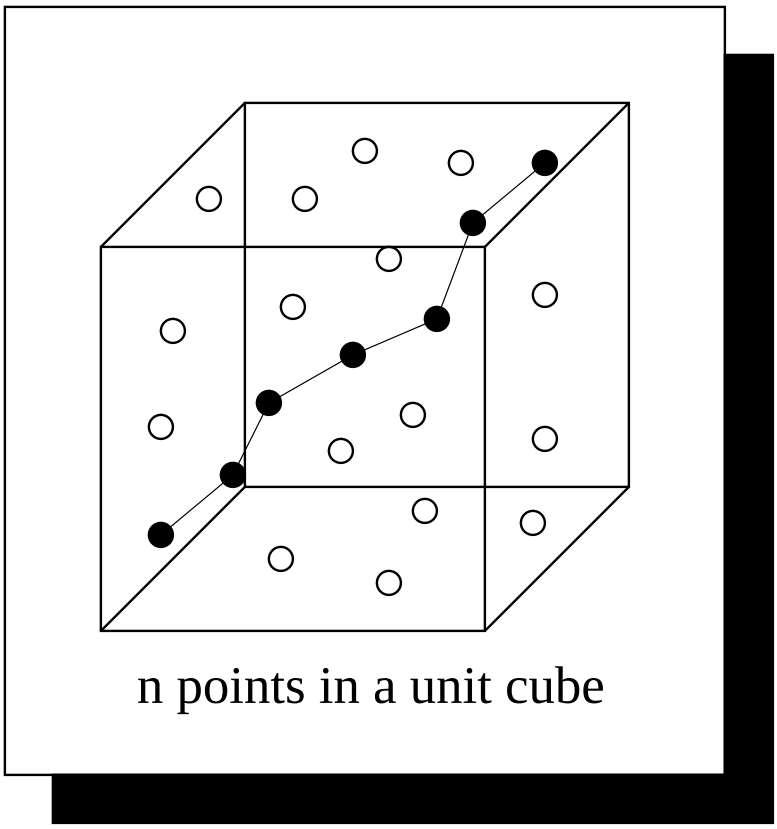
The Problem

Input

n random points in a d-dimensional unit cube, $\mathbf{V}_d = [0,1]^d$.

Output

- Size of the largest totally ordered subset,
- Length of the longest monotonically increasing subsequence, **or**
- Height, $H_d(n)$, of a maximal chain



$\mathbf{x}(i) \leq \mathbf{x}(j)$ if and only if $\mathbf{x}_k(i) \leq \mathbf{x}_k(j)$ for $k = 1, 2, \ldots, d$.

The Background

- **The problem** was posed by P. Erdös (1935);
- was used by Ulam to test the Monte Carlo method (1961); and
- was studied extensively by:
  - R. Baer (1968)
  - J. Hammersley (1972)
  - A. Verśik (1977)
  - B. Logan (1977)
  - S. Pilpel (1990)
  - J. Baik (1999)
  - A. Borodin (1999)
  - A. Odlyzko (2000)
- **The multi-dimensional problem** was initially considered by J. Steele (1977).

The Focus

$$c_d(n) = \frac{H_d(n)}{n^{1/d}}$$

What is Known:

- Bollabás and Winkler proved that $c_d(n)$ converges in probability to a constant c_d as n approaches infinity.
- $c_2 = 2$ was proven.
- $\lim_{d\to\infty} c_d = e$ was proven.

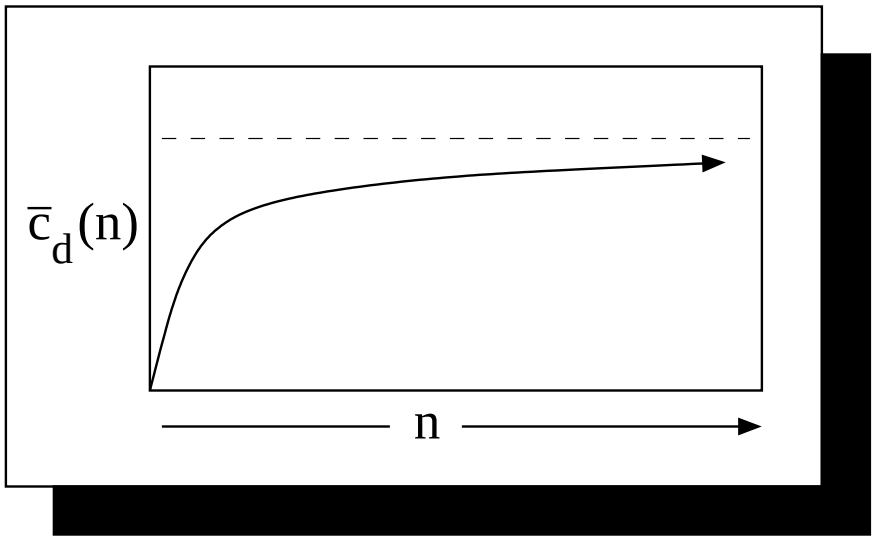
What is NOT Known:

- c_d for $d > 2$.
- Whether $\{c_d\}$ monotonically increases as d increases.

Our Goals

We would like to experimentally estimate c_d for various d's.

- Compute $\overline{c}_d(n)$ for very large n's.
- Show evidence that $\overline{c}_d(n)$ is converging as n increases.
- Show evidence that $\{\overline{c}_d(n)\}$ is increasing as d increases.



The Estimates

e_d is our estimate for c_d.

d	e_d
2	[1.998,2.002]
3	[2.363,2.366]
4	[2.514,2.521]
5	[2.583,2.589]
6	[2.607,2.617]

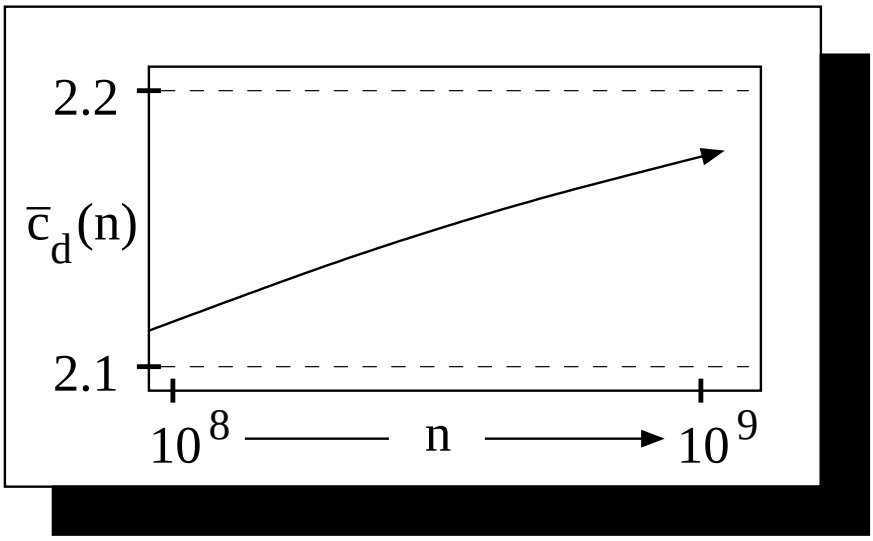
Recall that $c_2 = 2$ was theoretically proven.

The Challenges

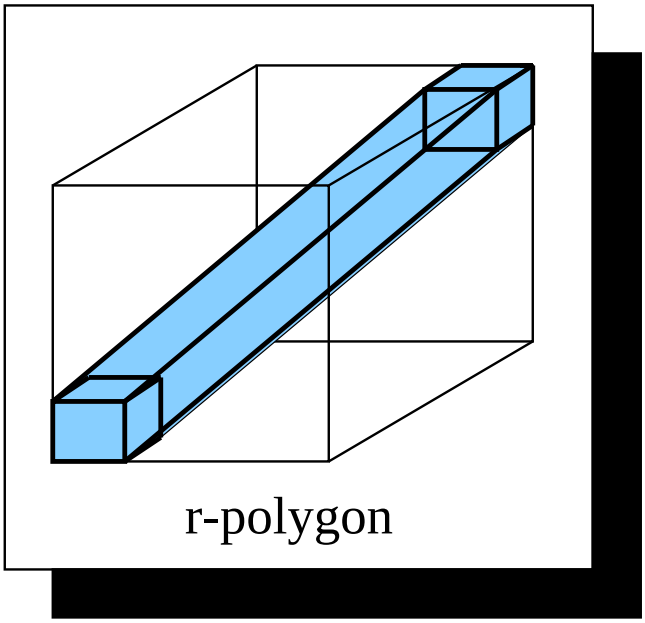
Complexity Computing the length of the longest chain requires $O(dn \log n)$ processing.

Observation Even for $d = 3$, $\overline{c}_d(n)$ shows no evidence of convergence for n up to 10^9.

Main Challenge Performing a sufficient number of trials for a sufficient n is not feasible.



Our Approach



r-polygon volume

$$V_d(r) = r^d + dr^{d-1}(1-r)$$

virtual input points

$$n^* = \lceil \frac{n}{V_d(r)} \rceil$$

If n^* is **very large**, we expect the longest chain to exist in the r-polygon and $\overline{c}_d(n, r)$ to approximate $\overline{c}_d(n^*)$

The Advantages

Boosting the virtual input size

- For small r,
  $n^* \gg n$.

- $c_d(n, r)$ may be computationally equivalent to $c_d(n^*)$, where n^* is infeasibly large.

Boosting the speed of the algorithm

- Since we only consider n points in an r-polygon, ordering properties can be used to discard many input points,

- which yields an $O(n^2rd)$-time $O(nrd)$-space algorithm,

- where $r \ll 1$ significantly boosts the algorithm's efficiency.

Note: There exists an $O(dn \log n)$-time algorithm.
However, at the time the paper was written,
it was believed that r could be decreased arbitrarily.
Thus, our approach could be more efficient in practice.

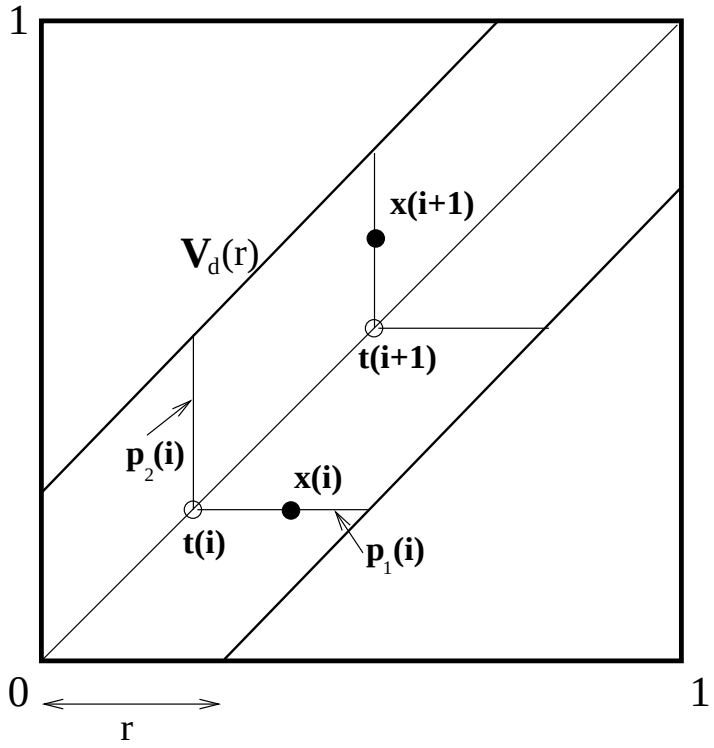
The Algorithm

Main features

1. Generate random input points "sequentially" within the r-polygon guaranteeing certain ordering properties.

2. Compute the length of the maximal chain as each input point is generated and use the ordering properties to discard "unneeded" input points.

Input Generation



Ordering Properties

1. If $\mathbf{t}(i) \geq \mathbf{x}(j)$, then, for all $k \geq i$, $\mathbf{x}(k) \geq \mathbf{x}(j)$

2. If $i < j$, it cannot be that $\mathbf{x}(i) \geq \mathbf{x}(j)$

Algorithm 1 (Sequential generation of input points)

1. Set $i = 1$ and $t_{prev} = 0$.

2. Generate the i^{th} origin point $\mathbf{t}(i) = (t(i), t(i), \ldots, t(i))$
   where $t(i) \in [t_{prev}, 1]$ is the i^{th} order statistic of n points.

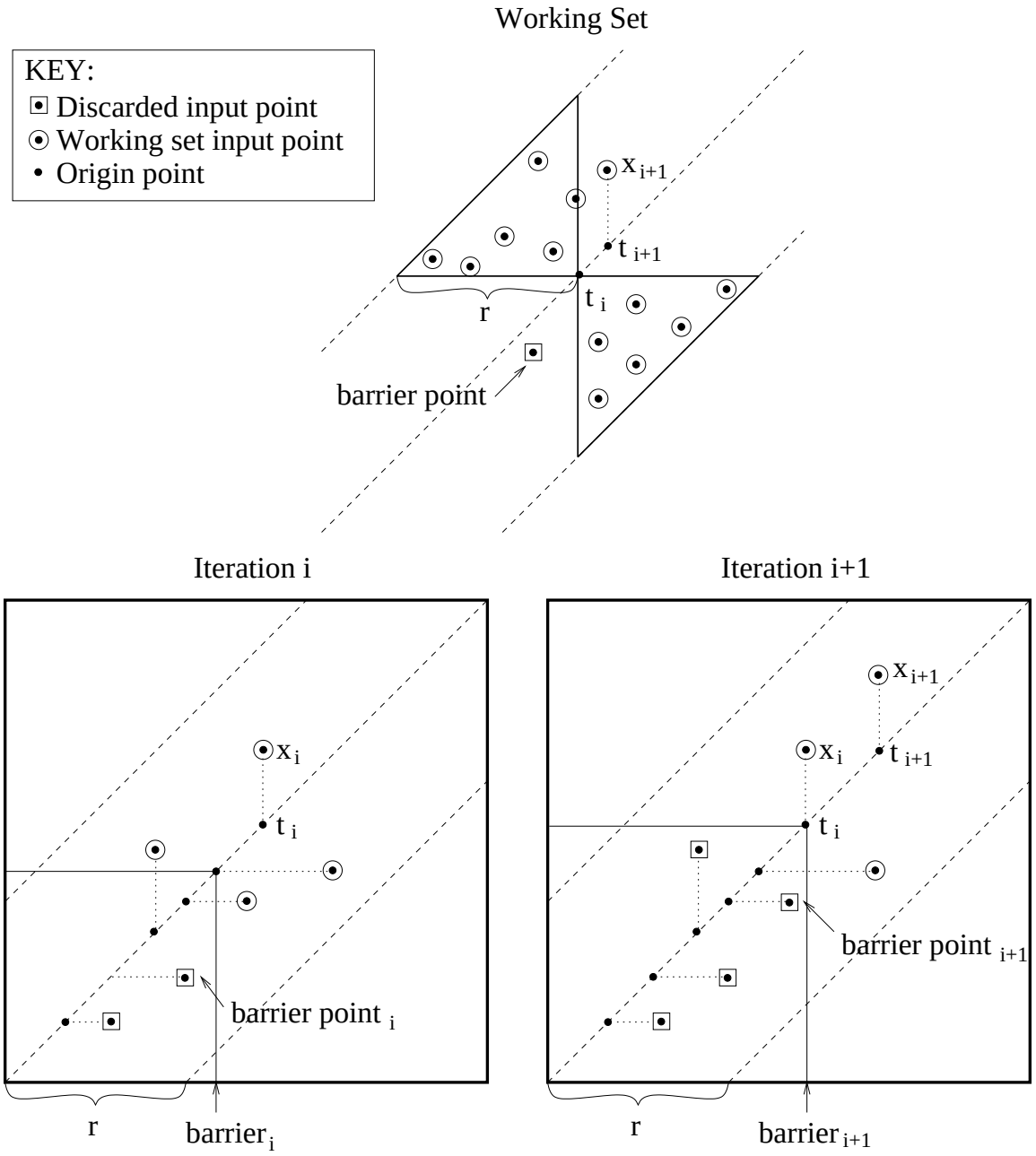
3. Generate a random vector $\mathbf{v} = \{v_1, v_2, \ldots, v_d\}$
   where each v_i is in $[0, \min\{r, 1 - t_i\}]$

4. Generate a random integer k from $\{1, 2, \ldots, d\}$
   and set $v_k = 0$

5. Generate input point $\mathbf{x}(i) = \{t(i) + v_1, t(i) + v_2, \ldots, t(i) + v_d\}$.

6. Set $t_{prev} = t(i)$, $i = i + 1$ and go back to step 2 if $i \leq n$.

Working Set Algorithm



Algorithm 2 (Computing the height)

1. Generate a new barrier $\mathbf{t}(i)$ and input point $\mathbf{x}(i)$.

2. For every point $\mathbf{x}(j)$ in the working set

   (a) Check if $\mathbf{x}(i)$ is above $\mathbf{x}(j)$ and update $\mathbf{x}(i)$'s height accordingly

   (b) If $\mathbf{x}(j)$ is below $\mathbf{t}(i)$ discard it unless its height is maximal among all discarded points.

3. Output the largest computed height.

The Disadvantage

Expectation

By continually decreasing r, we hope to boost the virtual input to sufficient size to achieve evidence of convergence.

Reality

- $c_d(n, r)$ is not computationally equivalent to $c_d(n^*)$ for every combination of n and r.
- In order to preserve computational equivalence, as r decreases, it is necessary to increase n.
- Once again, n must be increased to an infeasible size.

A New Approach

- Another approach to estimating c_d is to select a set $\{r_i\}_{i=1}^{T}$ of values for r and consider T sequences $\{\overline{c}_d(n, r_i)\}$ for $i = 1 \ldots T$.

- Each of these sequences converges to the same limit c_d as $n \to \infty$.

- The problem now is one of estimating the common limit of these T sequences.

Co-convergence

Based on the following **assumption**

$$\forall r > 0,\ \overline{c}_d(n, r) = c_d - \mu(r) f(n) + o(f(n))$$

we can arrive at the following **relationship** for any two $c_d(n, r)$:

$$\overline{c}_d(n, r_i) = (1 - A(r_i, r_j))\, c_d + A(r_i, r_j)\overline{c}_d(n, r_j) + o(f(n))$$

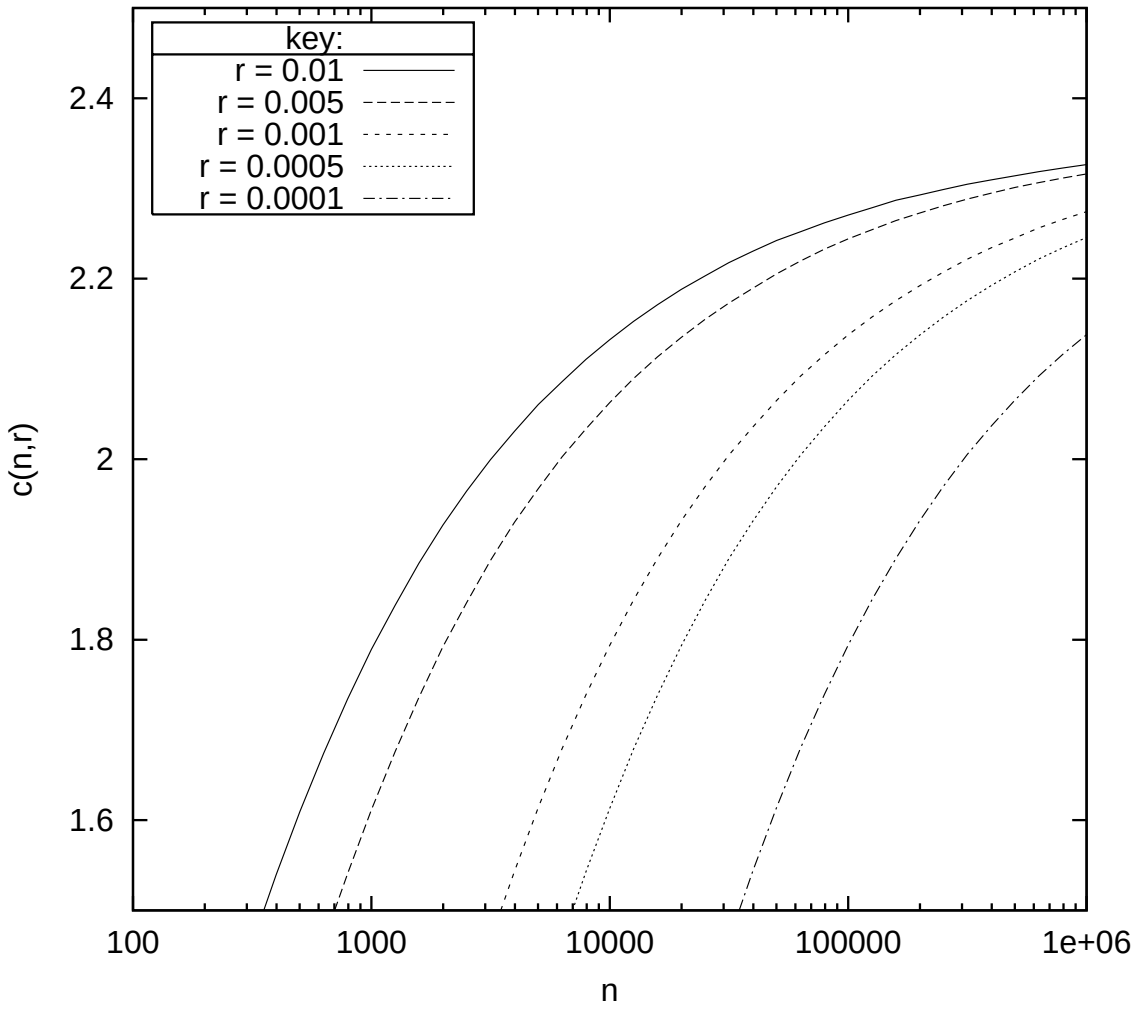
then we can derive **estimates** , which are themselves sequences, for each pair of r's.

$$c_d(n, r_i, r_j) = B_d(n, r_i, r_j)/(1 - A_d(n, r_i, r_j))$$

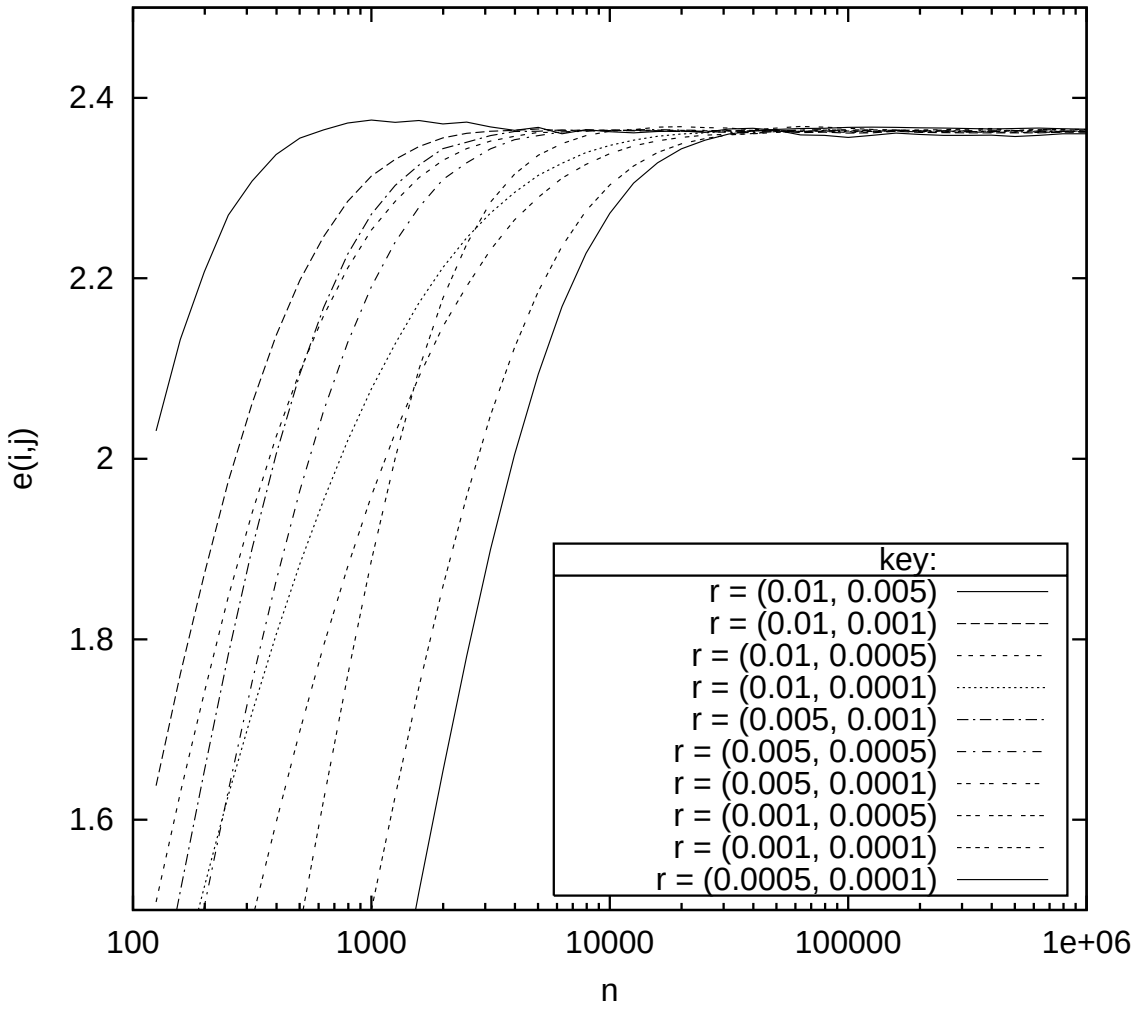
Expectations

- If our assumption is correct, every estimate (for every pair of r's) should converge to the same value.
- Our hope is that each $c_d(n, r_i, r_j)$ converges faster than their respective $c_d(n, r_i)$ and $c_d(n, r_j)$.

Convergence of $c_3(n, r)$



Co-convergence of $c_3(n, r_i, r_j)$



The Estimates

e_d is the range for which $c_d(n, r_i, r_j)$ converged for each pair (r_i, r_j)

d	e_d
2	[1.998,2.002]
3	[2.363,2.366]
4	[2.514,2.521]
5	[2.583,2.589]
6	[2.607,2.617]

Recall that $c_2 = 2$ was theoretically proven.