The flow limit of reflect-reflect-relax

Veit Elser
Department of Physics
Cornell

Dedicated to the memory of Jon Borwein

Hi Jon,

I'm on sabbatical at Stanford and have been talking with Stephen Boyd.

Do you have any experience with a skewed version of the **reflect-reflect-average algorithm**?

x' = (1-beta) x + beta R_A(R_B(x))

Having **beta<1/2 appears to really help** in some problems.

-Veit

May 14, 2015

Dear Veit.
Do send my best to Steve.
In general for projection and reflection methods, especially in the convex case, **under-relaxation can improve convergence theory** while slowing things down.

I would be interested in knowing more about your examples where it is helping?

Cheers, Jon

Outline

- RRR and hard problems
- first evidence: bit retrieval
- flow limit
- more evidence: latin squares
- interpretation

reflect-reflect-relax (RRR)

$$x \mapsto (1-\beta/2)x + (\beta/2)R_2(R_1(x))$$

$$R_1(x) : \text{reflect } x \text{ through constraint set } 1$$

$$R_2(x) : \text{reflect } x \text{ through constraint set } 2$$

$$\beta = 1 : \text{Douglas-Rachford}$$

$$x \mapsto x + \beta\,(P_2(2P_1(x) - x) - P_1(x))$$

$$P_1(x) : \text{project } x \text{ to constraint set } 1$$

$$P_2(x) : \text{project } x \text{ to constraint set } 2$$

RRR on hard feasibility problems

- at least one constraint set is non-convex, often discrete
- use RRR to generate **samples**
- not interested in convergence in usual sense
- can round to discrete set to verify solutions
- comparison group: combinatorial sampling/search algorithms

Does RRR sampling offer advantages
over standard methods?

first application

binary (two-valued) sequence: $+1 \quad -1 \quad -1 \quad +1 \quad -1$

periodic autocorrelations: $+5 \quad -3 \quad +1 \quad +1 \quad -3$

$$a_i = \sum_{j=0}^{N-1} s_j s_{j-i}$$

bit retrieval:

reconstruct a binary sequence from its periodic autocorrelations

medium difficulty instance

a_i :	101	−19	13	−3	9	−7	13	−7	−23	−3	−27
		17	−11	5	−23	17	1	9	−3	9	−3
		1	13	1	−11	−7	−3	−11	21	−19	−15
		−11	1	−11	17	−3	1	1	1	21	−3
		13	−11	9	−11	13	−11	13	−7	−3	1
		1	−3	−7	13	−11	13	−11	9	−11	13
		−3	21	1	1	1	−3	17	−11	1	−11
		−15	−19	21	−11	−3	−7	−11	1	13	1
		−3	9	−3	9	1	17	−23	5	−11	17
		−27	−3	−23	−7	13	−7	9	−3	13	−19

fastest known algorithm: first factor this integer

520964398733896867126337878104465216436493073798791761571444565192819298905650746097664

completely defeated by noise: $a_i \to a_i \pm 2$

complexity of retrieving *N* bits

- complexity class unknown
- circumstantially hard: basis of crypto schemes
- best noise-free algorithm needs to factor (*N* log *N*)-bit integer
- minimal noise, preserves solution uniqueness: $a_i \to a_i \pm 2$
- best algorithms for noisy bit retrieval: complexity 2^{cN}

c	branch & bound	RRR
average case	0.36	0.21
worst case	0.56	0.50

V. Elser, The complexity of bit retrieval, IEEE Trans. Info. Theory, to appear
arXiv:1601.03428

bit retrieval with RRR

P_1 : projection to sequence of signs

P_2 : projection to real sequence with given autocorrelation

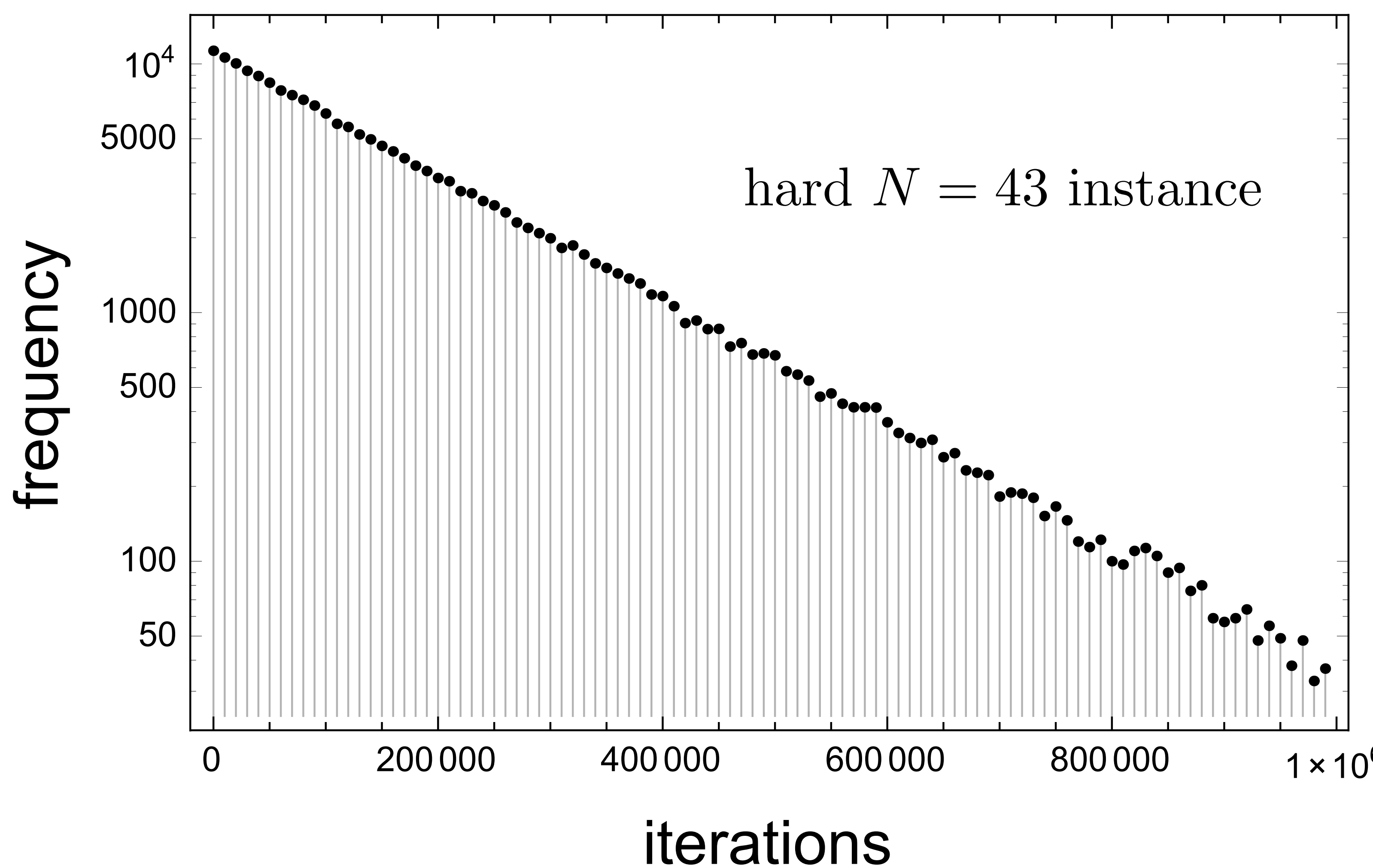
$$x \leftarrow \text{ random initial sequence}$$

$$\text{iterate: } x \leftarrow \mathrm{RRR}(x)$$

$$\text{terminate: } P_1(x) \text{ has given autocorrelation}$$

experimental results

- 100% success rate
- exponential-decay distribution of iteration counts



interpretation

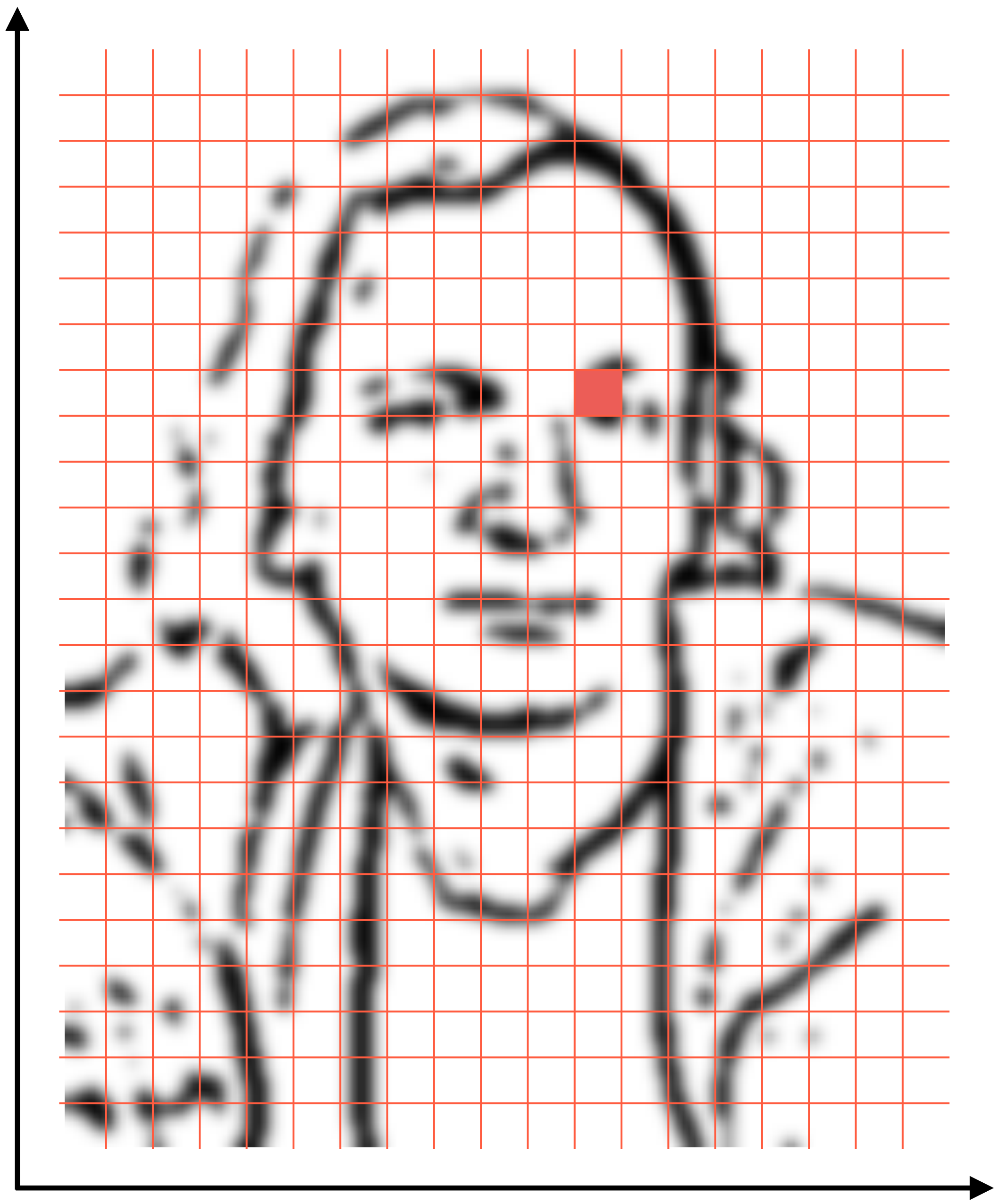
infeasible case:

The sequence x_0, x_1, x_2, … of RRR iterates samples a probability density on $\mathbb{R}^N$, independent of the initial point.

interpretation

feasible case:

The size of the fixed-point basin determines the expected number of RRR iterations.



mixing dynamics:

- few (ideally one) attractors

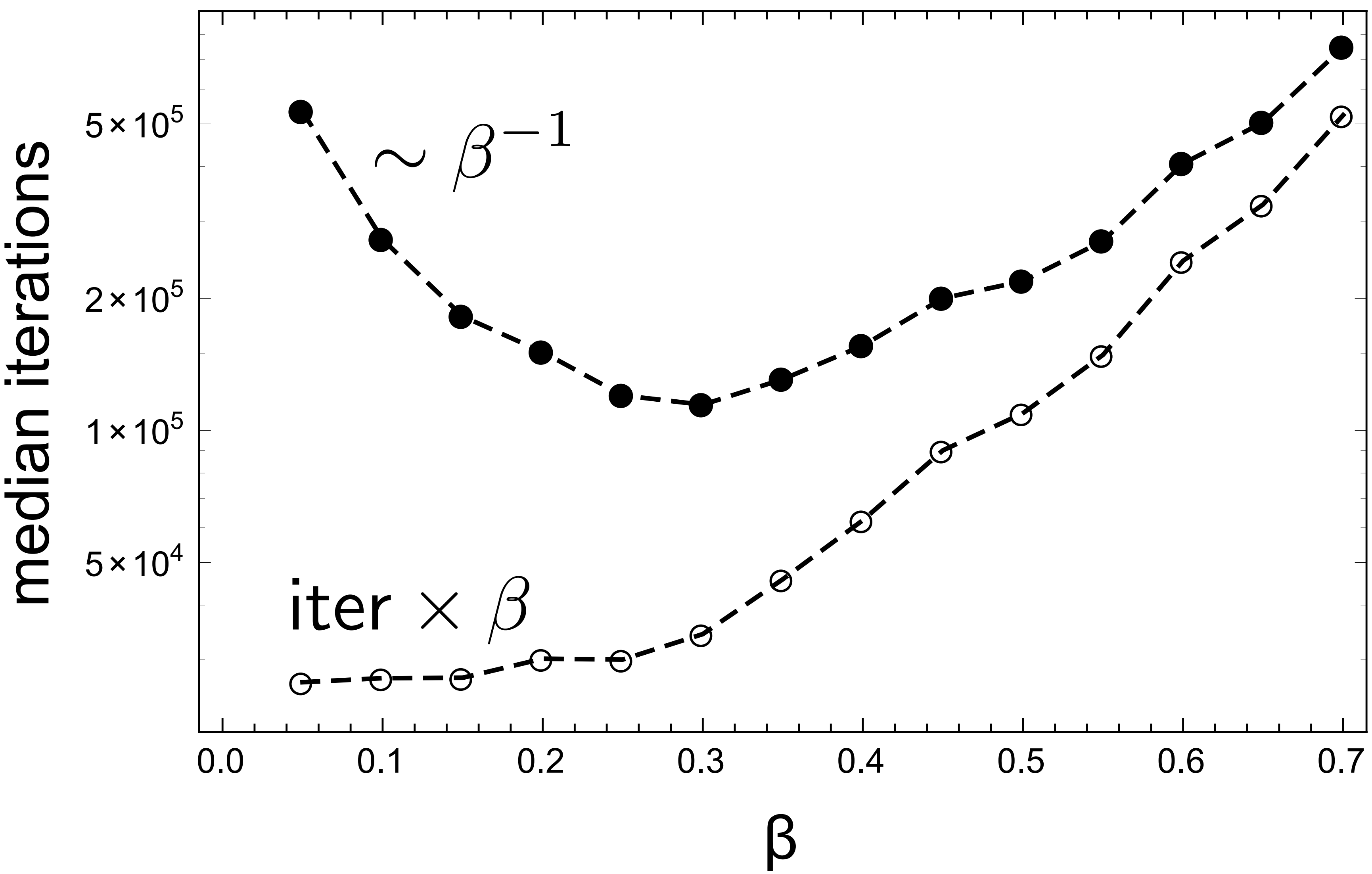
optimize algorithm:

- decrease size* of attractor
- increase size of fixed-point basin

We have one parameter: β

* Kolmogorov-Sinai entropy

more experimental results



$\text{iter} \times \beta = \text{'time' elapsed to find solution when 'timestep' is } \beta$

flow limit

$$x_{n+1} = x_n + \Delta\beta \left(P_2(2P_1(x_n) - x_n) - P_1(x_n)\right)$$

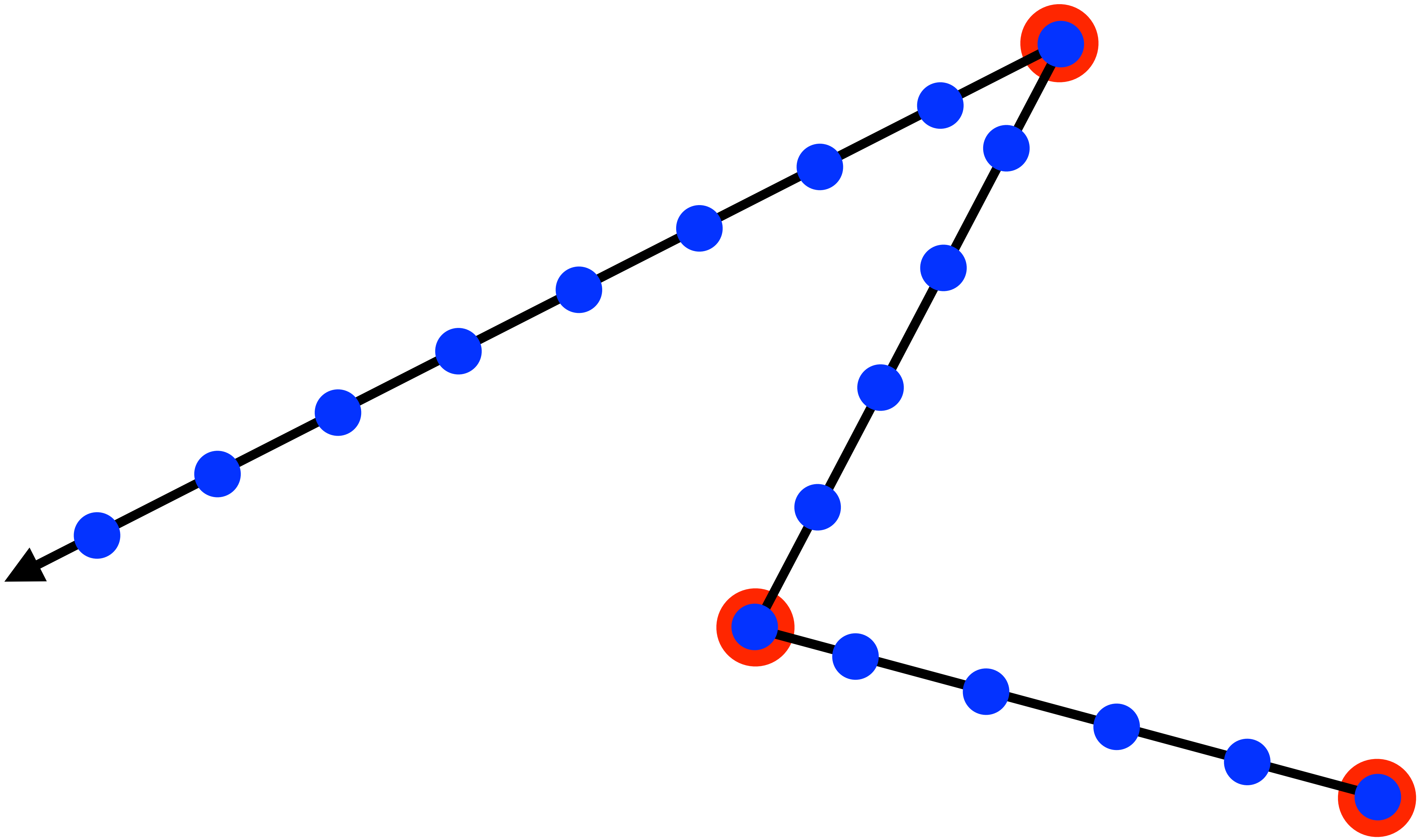
$$\Delta\beta \to 0 : \qquad \frac{x_{n+1} - x_n}{\Delta\beta} \to \dot{x}(\beta)$$

$$\dot{x} = P_2(2P_1(x) - x) - P_1(x)$$

RHS = flow field

special case:

- discrete constraint sets
- piecewise-constant flow field

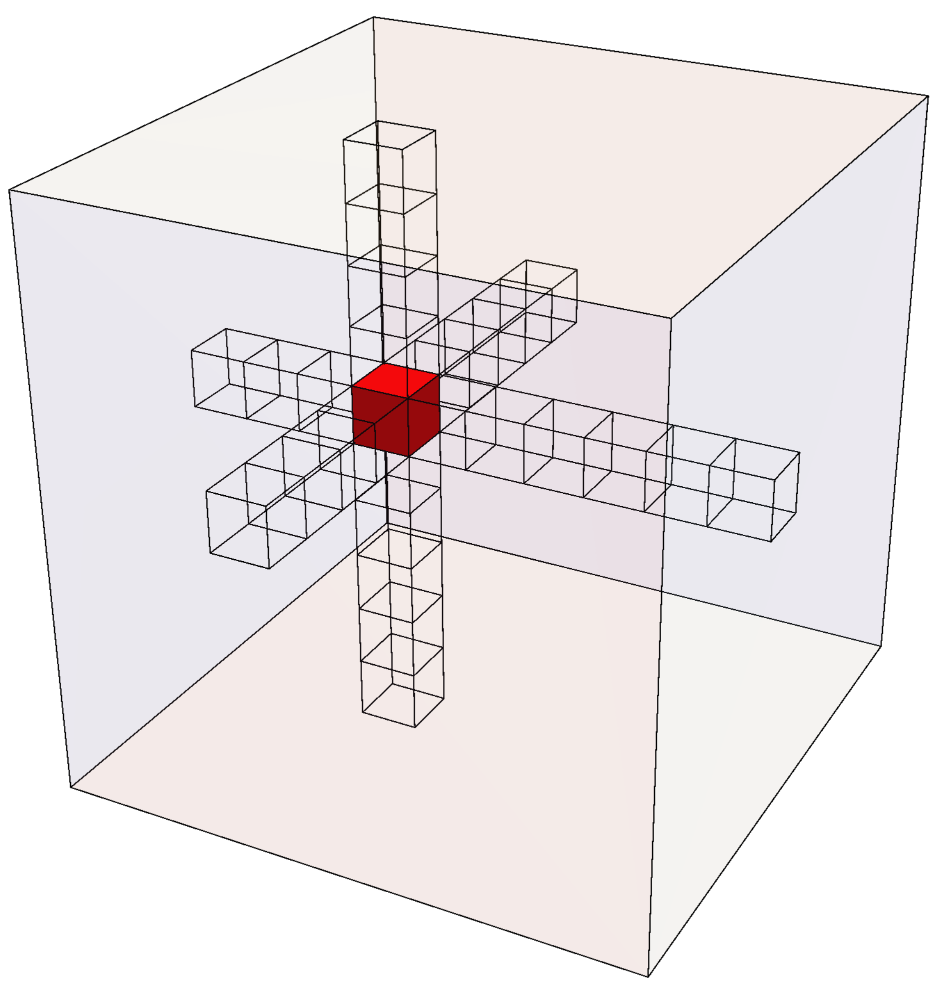


Can devise, in principle, iteration scheme based on Voronoi cells of constraint sets.

more evidence: latin square completion

a		c	
		d	
	b		
d			a

a	d	c	b
b	a	d	c
c	b	a	d
d	c	b	a



binary encoding:

exactly one 1 in every stack
in all three dimensions

divide and concur formulation

three blocks of n^3 numbers for independent (divided) constraints

block 1: x-stacks
block 2: y-stacks
block 3: z-stacks

divide constraint:

- single 1 in each x-stack of block 1
- single 1 in each y-stack of block 2
- single 1 in each z-stack of block 3

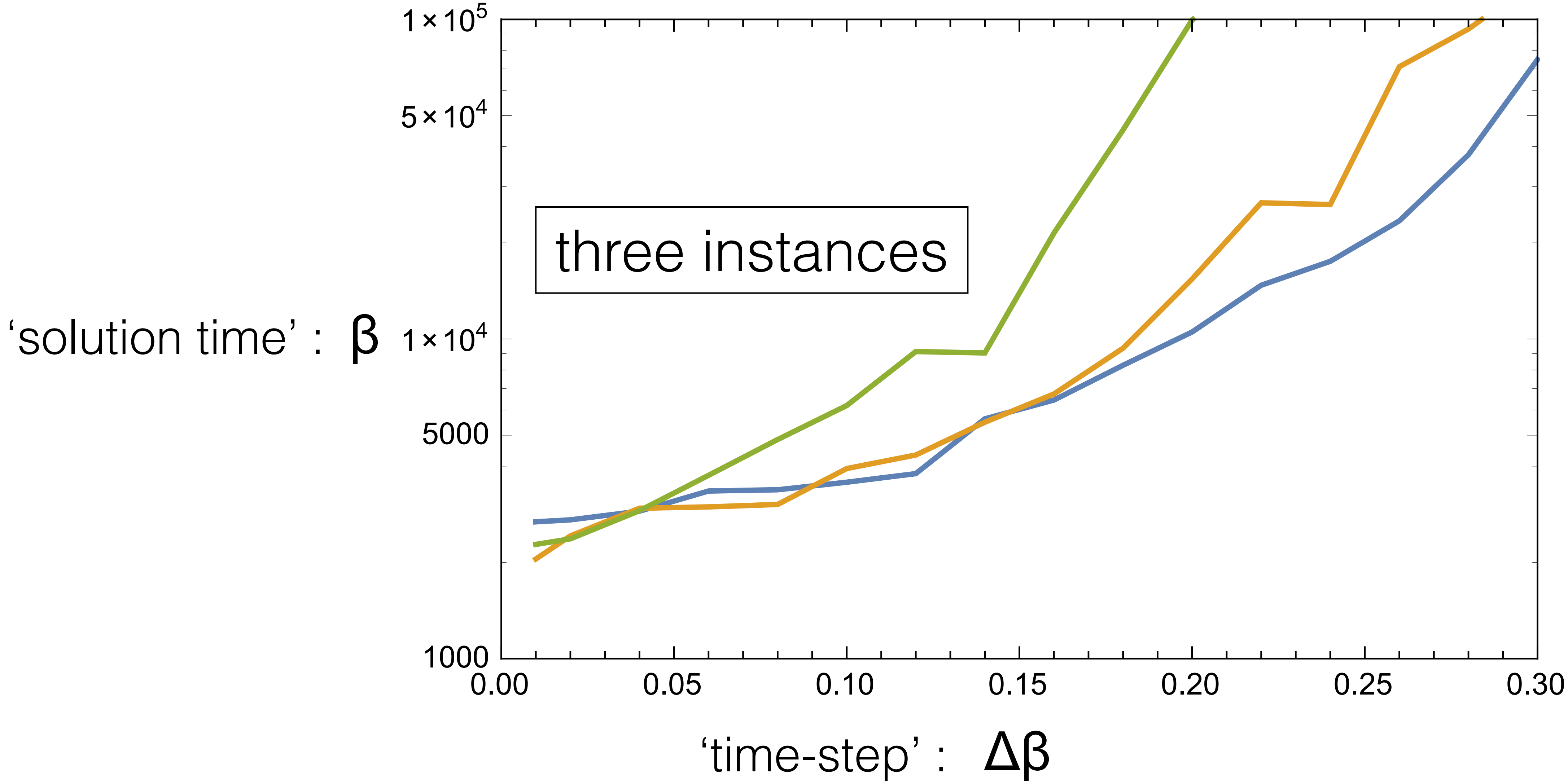
concur constraint:

- block 1 = block 2 = block 3
- all elements 0 or 1

both constraints are discrete

latin square completion results

25 x 25, 65% complete



same behavior as bit retrieval

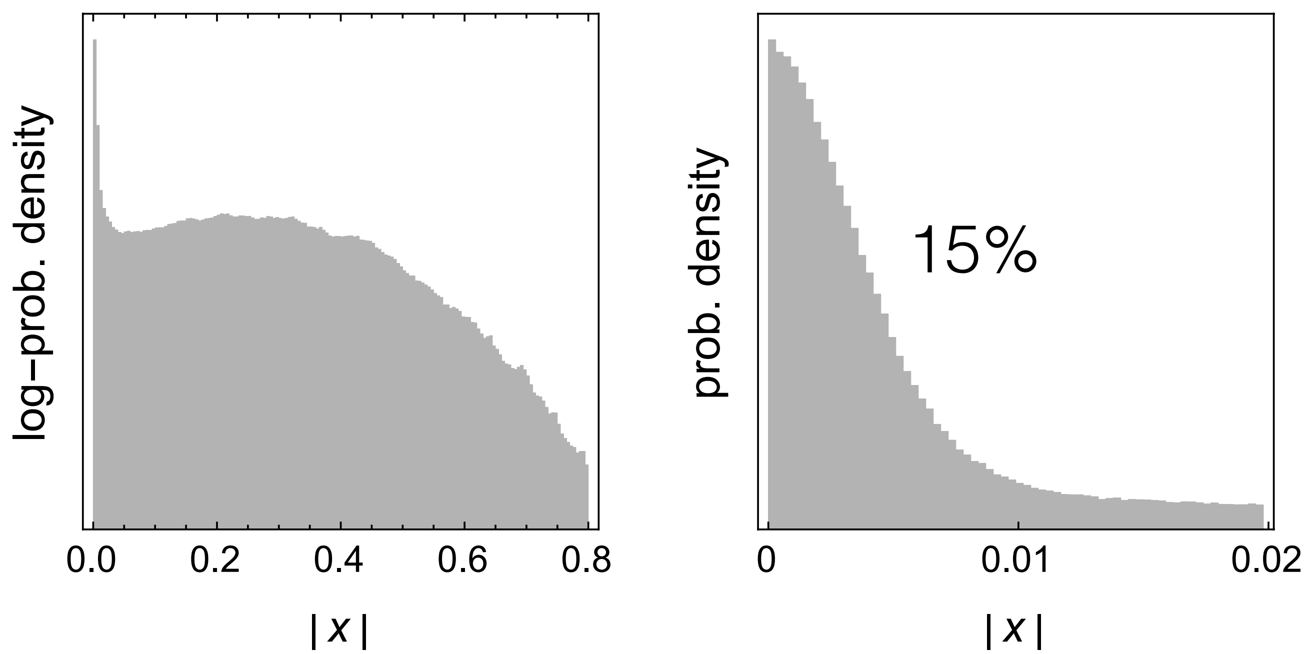
two questions

1. Is RRR doing something special in the flow limit?

2. How does the special behavior improve RRR search?

flow limit behavior in bit retrieval

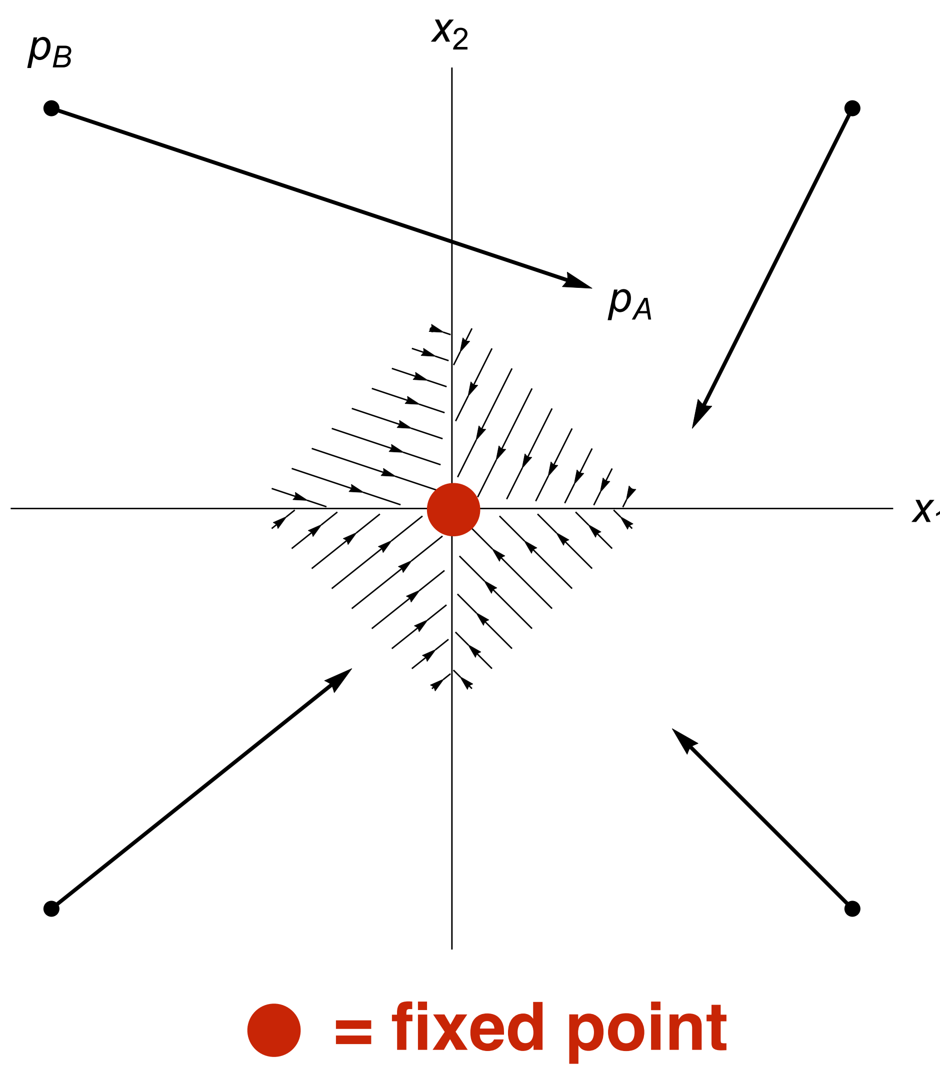
$$\Delta\beta = 0.01$$



search vector components concentrated at zero

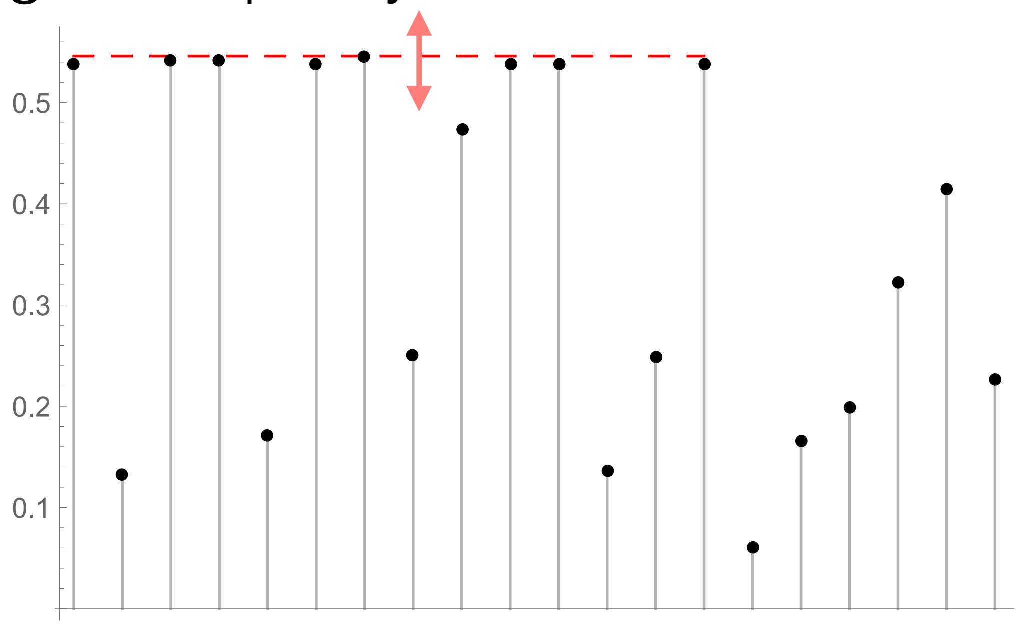
flow is confined to Voronoi facets of all-signs constraint set

facet attraction mechanism



flow limit behavior in latin square completion

high multiplicity of maximum elements



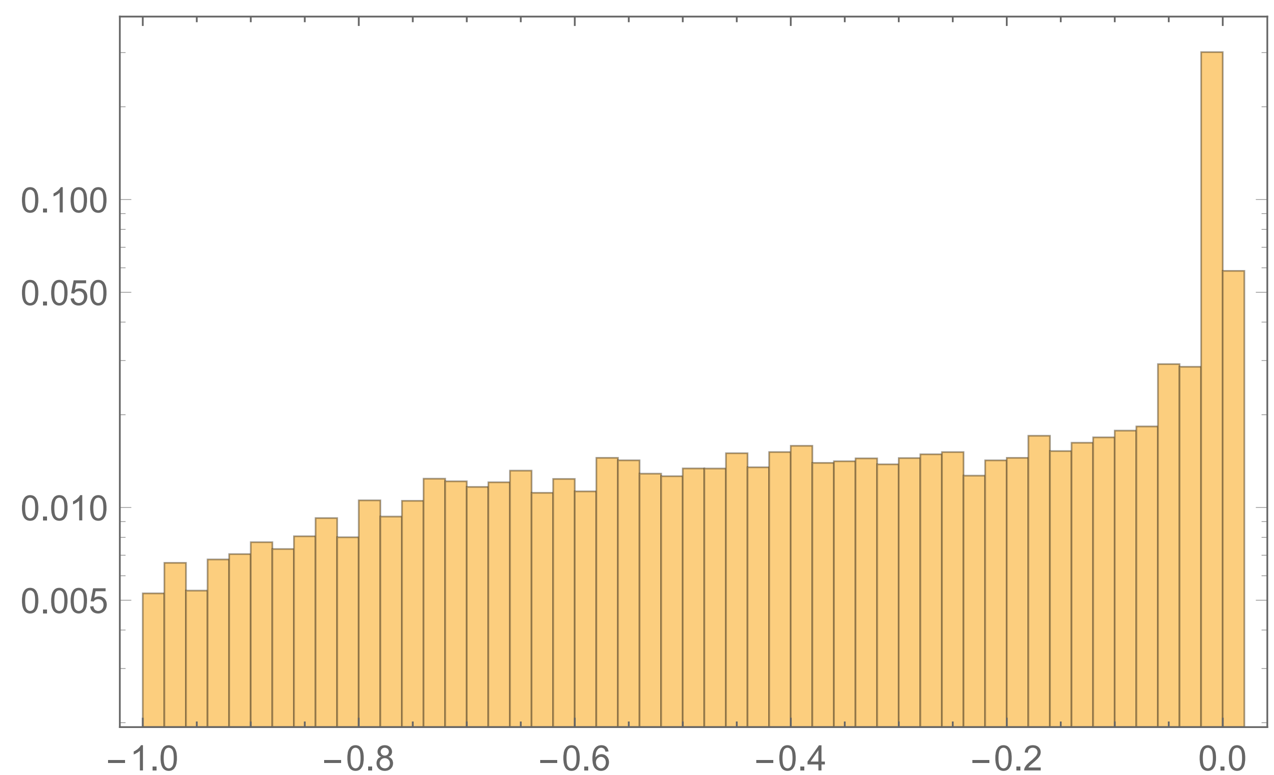
$$\begin{array}{ccccc} 0 & 0\,0 & 0\,1 & 0\,0 & 0 \\ 0 & 1\,0 & 0\,0 & 0\,0 & 0 \\ 0 & 0\,1 & 0\,0 & 0\,0 & 0 \end{array} \quad P_1$$

$$\cdots$$

$$\bar{P}_1 = \frac{1}{8}$$

$$\dot{x} = P_2 - \bar{P}_1 = \begin{cases} 0 - \frac{1}{8} \\ 1 - \frac{1}{8} \end{cases}$$

distribution of elements in a stack relative to its maximum



$$x - \max_{\text{stack}}(x)$$

A working hypothesis:

The flow limit improves search because the entropy of the RRR attractor is reduced.

$\beta \to 0$