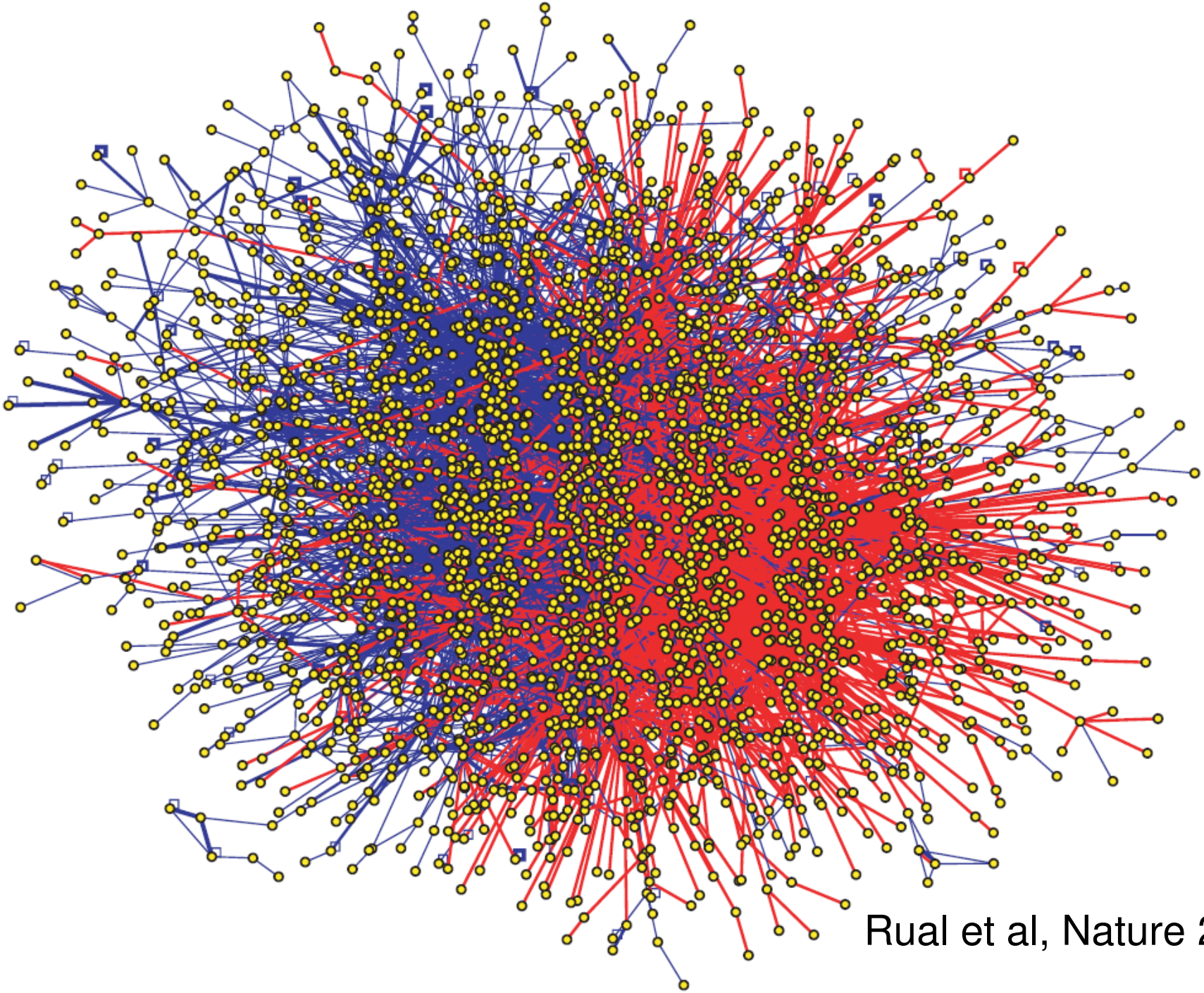
Shifted Transversal Design smart-pooling for high-throughput biology

Nicolas Thierry-Mieg
CNRS / TIMC-IMAG / BCM, Grenoble, France



Rual et al, Nature 2005

CCSB-HI1

Assay: yeast two-hybrid (Y2H)

Space: 8100x8100

2800 interactions (1 / 23K pairs)

125 retested by co-AP: ~80% success

-> few (technical) false positives, but many false negatives

Screen-Seq protocol:

- one bait against mini-pools of 188 preys, 96-well format
- identification by sequencing
- pairwise retests

Smart-pooling == CGT

Y2H and many other HT experiments:

- basic **yes-or-no test** to a large collection of "objects"
- **low-frequency** positives
- **experimental noise**

Smart-pooling: increase **efficiency**, **accuracy** & **coverage**

provided that

- objects individually available (eg ORFeome)
- basic assay works on pools (logical OR)
- Cherry-picking robot

Smart-pooling == CGT

Y2H and many other HT experiments:

- basic **yes-or-no test** to a large collection of "objects"
- **low-frequency** positives
- **experimental noise**

Smart-pooling: increase **efficiency**, **accuracy** & **coverage**

provided that

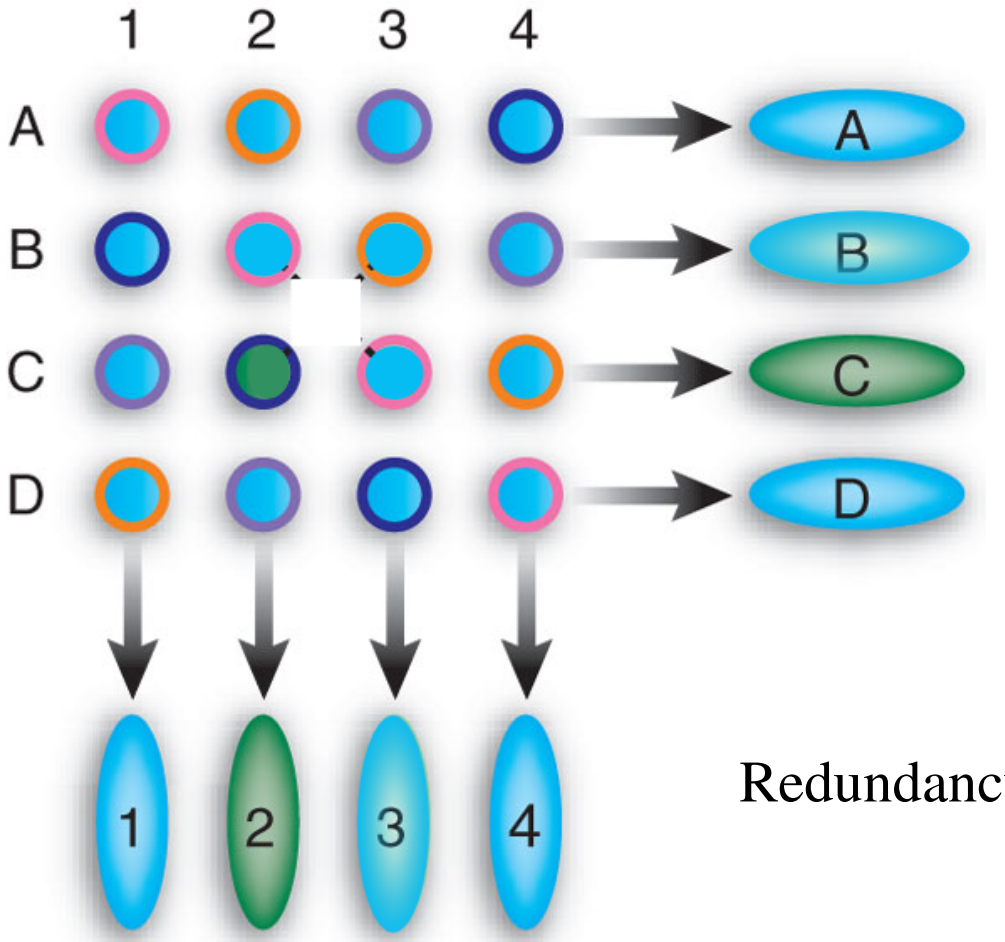
- objects individually available (eg ORFeome)
- basic assay works on pools (logical OR)
- Cherry-picking robot... **or Stefano!**

Smart-pooling

General idea:

- **small number** of **redundant** pools
- **direct identification** from the pattern of positive pools (no sequencing in Y2H)
- deal with **false positives & negatives**

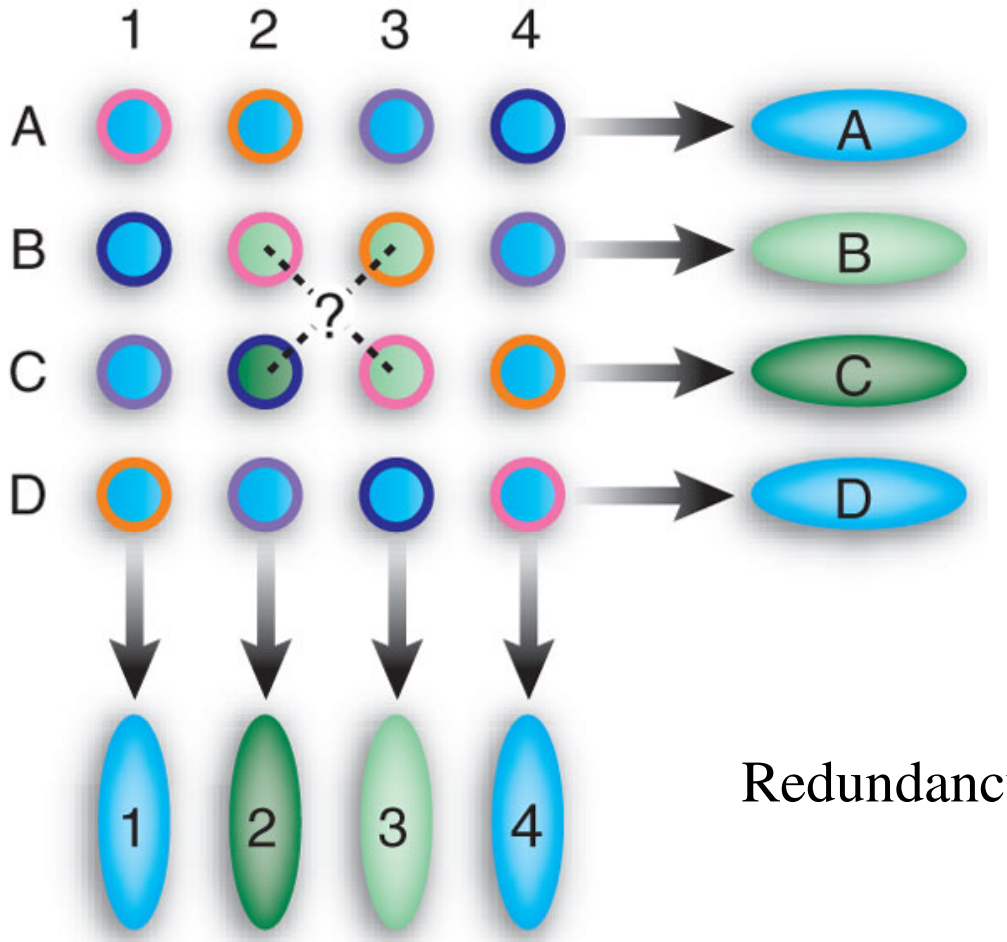
Rows-and-columns design



Redundancy 2

Thierry-Mieg, Nat Methods 2006

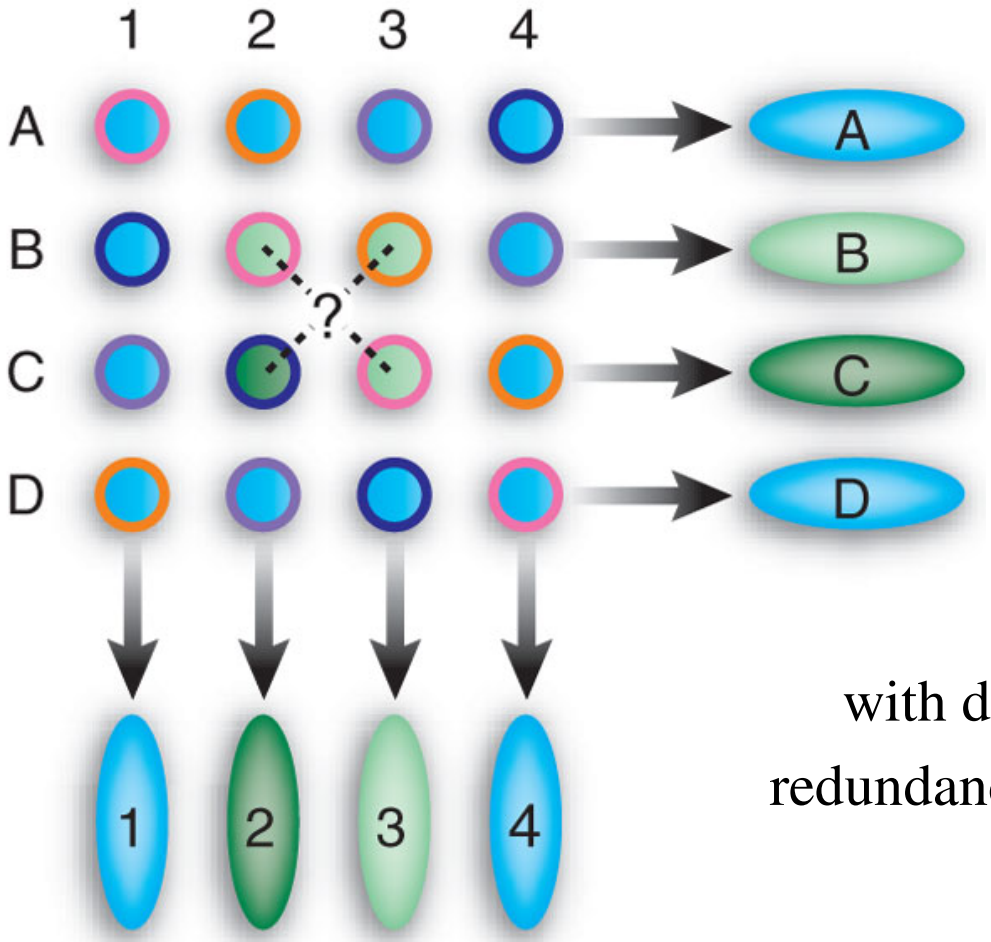
Rows-and-columns design



Redundancy 2, extra redundancy 0

Thierry-Mieg, Nat Methods 2006

Rows-and-columns design



with diagonal pools:

redundancy **3**, extra redundancy **1**

Thierry-Mieg, Nat Methods 2006

Talk layout

- **The pooling problem: how to design pools?**
- The decoding problem: how to interpret results?
- Experiments: Y2H interactome mapping
  - Human: 100 baits vs 940 preys
  - Worm: 12 baits vs 13000 preys

The pooling problem

Pooling problem (Combinatorial Group Testing problem) (n,t,E):

- $\mathcal{A}_n$ a set of Boolean variables (n≈ 100-10^4)
- t = max number of positives (≈ 1-10)
- E = max number of errors (≈ 1-40% of tests)

Pool: subset of $\mathcal{A}_n$, value=OR

Goal: build a set of **v** pools

- v as small as possible
- guarantee correction of errors & identification of positives

Shifted Transversal Design: idea

"Transversal" construction: layers

"Shift" variables from layer to layer

- Limit co-occurrence of variables
- Constant-sized intersections between pools

Shifted Transversal Design: idea

"Transversal" construction: layers

"Shift" variables from layer to layer

- Limit co-occurrence of variables
- Constant-sized intersections between pools

STD(n;q;k) :

- **n** variables
- **q** prime, $q < n$
- **k** = number of layers ($k \leq q$)

Each layer: symmetric construction, q pools of size n/q or 1+n/q

Matrix representation

v× n Boolean matrix: M(i,j) true ⇔ pool i contains variable j

Example: n=9, $\mathcal{A}_9 = \{0, 1, \ldots, 8\}$:

									pools
1	0	0	1	0	0	1	0	0	{0, 3, 6}
0	1	0	0	1	0	0	1	0	{1, 4, 7}
0	0	1	0	0	1	0	0	1	{2, 5, 8}

"layer" = partition of $\mathcal{A}_n$

STD construction

Let:

- σ_q circular permutation on $\{0,1\}^q$: $\sigma_q \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_q \end{bmatrix} = \begin{bmatrix} x_q \\ x_1 \\ \vdots \\ x_{q-1} \end{bmatrix}$

- $\Gamma(q,n) = \min\{\gamma \mid q^{\gamma+1} \geq n\}$

Example: n=940, q=13 => Γ=2

STD construction

$\forall$ j $\in$ {0,...,q-1}: M_j qxn Boolean matrix, representing layer L(j)

columns $C_{j,0}, \ldots, C_{j,n-1}$:

$$C_{0,0} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

STD construction

$\forall$ j $\in \{0,...,q\text{-}1\}$: M_j qxn Boolean matrix, representing layer L(j)

columns $C_{j,0}, \ldots, C_{j,n-1}$:

$$C_{0,0} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \text{ and } \forall i \in \{0,\ldots,n\} \; C_{j,i} = \sigma_q^{s(i,j)}(C_{0,0}) \quad \text{where:}$$

$$s(i,j) = \sum_{c=0}^{\Gamma} j^c \left\lfloor \frac{i}{q^c} \right\rfloor$$

For k $\in \{1,...,q\}$, STD(n;q;k) = L(0) $\cup$ ... $\cup$ L(k-1)

STD construction

$\forall\, j \in \{0,...,q\text{-}1\}$: M_j qxn Boolean matrix, representing layer L(j)

columns $C_{j,0}, \ldots, C_{j,n-1}$:

$$C_{0,0} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \text{ and } \forall\, i \in \{0,\ldots,n\} \;\; C_{j,i} = \sigma_q^{s(i,j)}(C_{0,0}) \quad \text{where:}$$

$$s(i,j) = \sum_{c=0}^{\Gamma} j^c \left\lfloor \frac{i}{q^c} \right\rfloor$$

Strongly explicit

For $k \in \{1,...,q\}$, $\mathrm{STD}(n;q;k) = L(0) \cup \ldots \cup L(k\text{-}1)$

STD example: n=9, q=3

$$M_0 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

L(0) = {{0,3,6},{1,4,7},{2,5,8}}

j=0, hence $s(i,j) = \sum_{c=0}^{\Gamma} j^c \left\lfloor \frac{i}{q^c} \right\rfloor = i$

Column i has a 1 in row i mod q

In other words: shift the 1 downwards once to go from one column to the next (cycle at bottom)

STD example: n=9, q=3

$$M_0 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$L(0) = \{\{0,3,6\},\{1,4,7\},\{2,5,8\}\}$$

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$L(1) = \{\{0,5,7\},\{1,3,8\},\{2,4,6\}\}$$

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$L(2) = \{\{0,4,8\},\{1,5,6\},\{2,3,7\}\}$$

$$STD(n=9;q=3;k=2) = L(0) \cup L(1)$$

STD example: n=9 to 27, q=3

n=9, q=3, third layer (j=2): $M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$

n=27, q=3, j=2:

STD example: n=9 to 27, q=3

n=9, q=3, third layer (j=2): $M_2 =$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

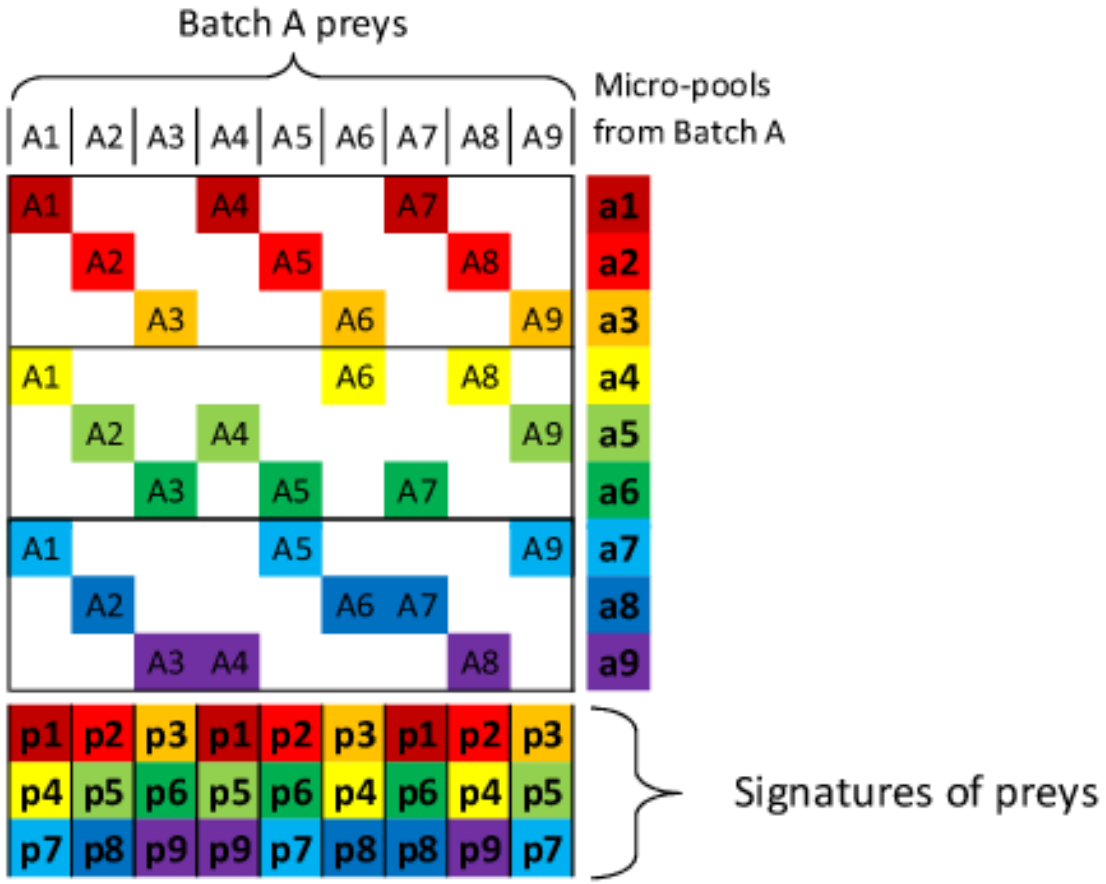
n=27, q=3, j=2: $s(i,j) = \sum_{c=0}^{\Gamma} j^c \left\lfloor \frac{i}{q^c} \right\rfloor = i + 2\left\lfloor \frac{i}{3} \right\rfloor + 4\left\lfloor \frac{i}{9} \right\rfloor$

+1 +(1+j) +(1+j+j^2)

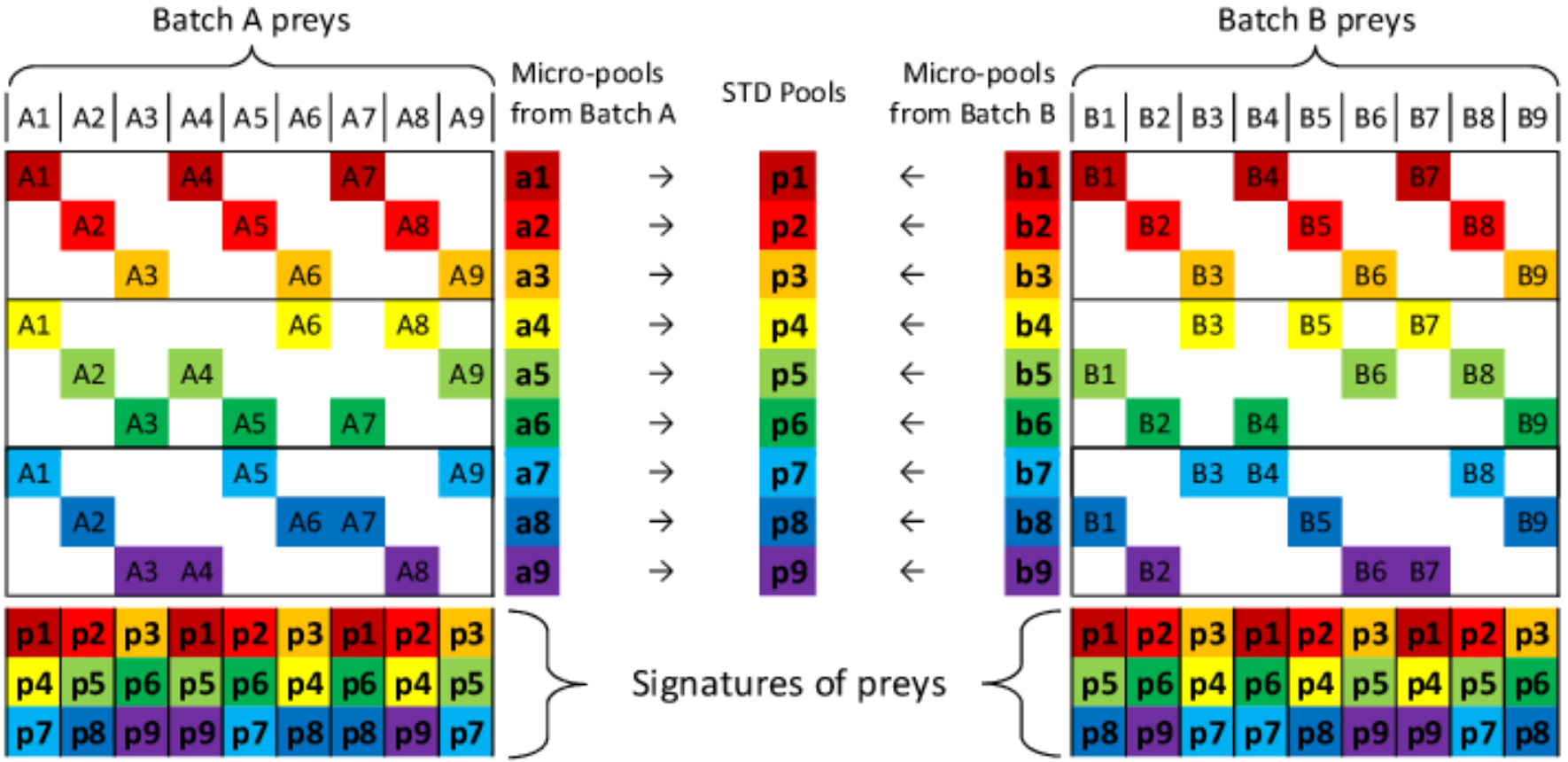
$M_2 =$

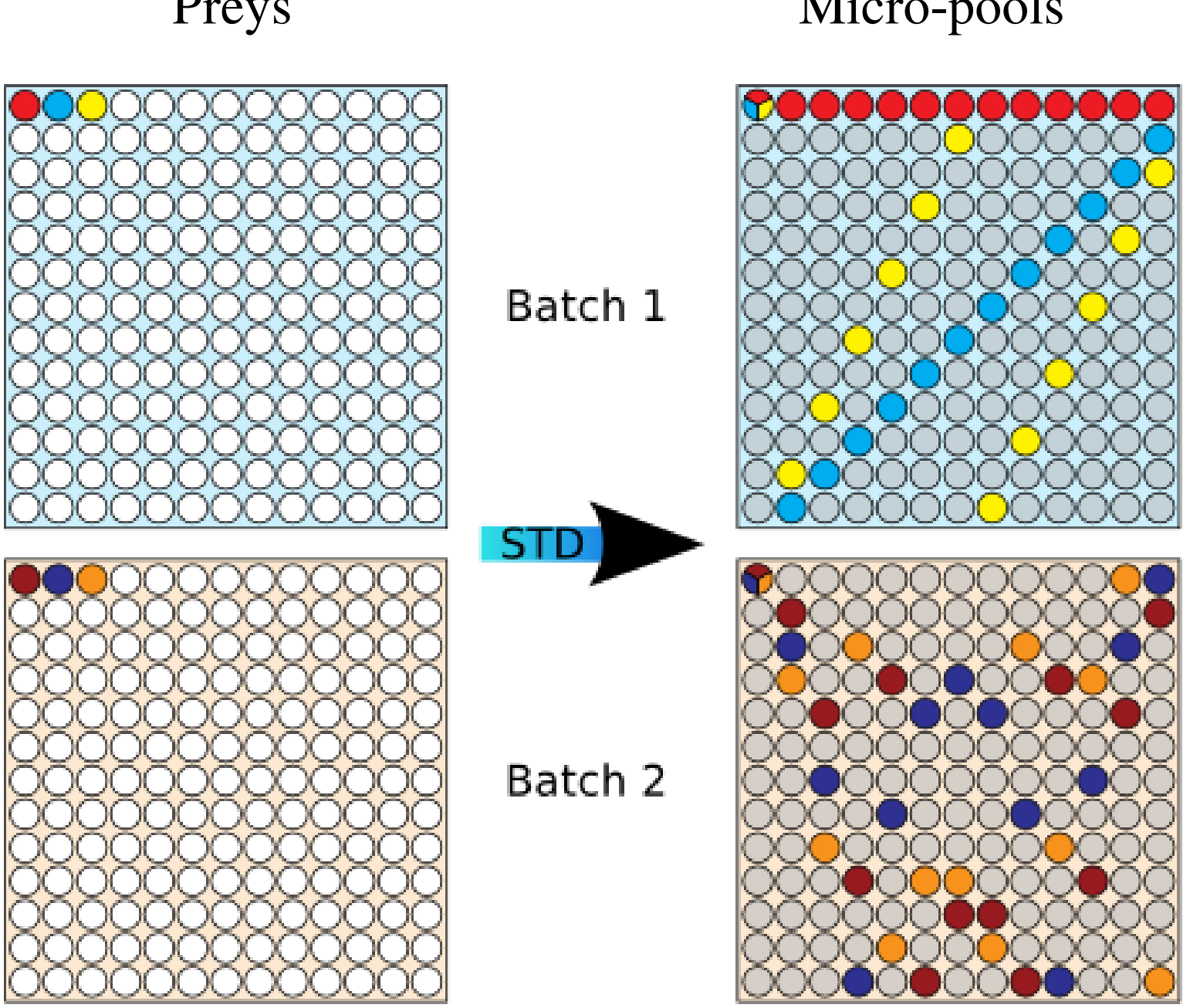
$$\left[\begin{array}{ccccccccc|ccccccccc|ccccccccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

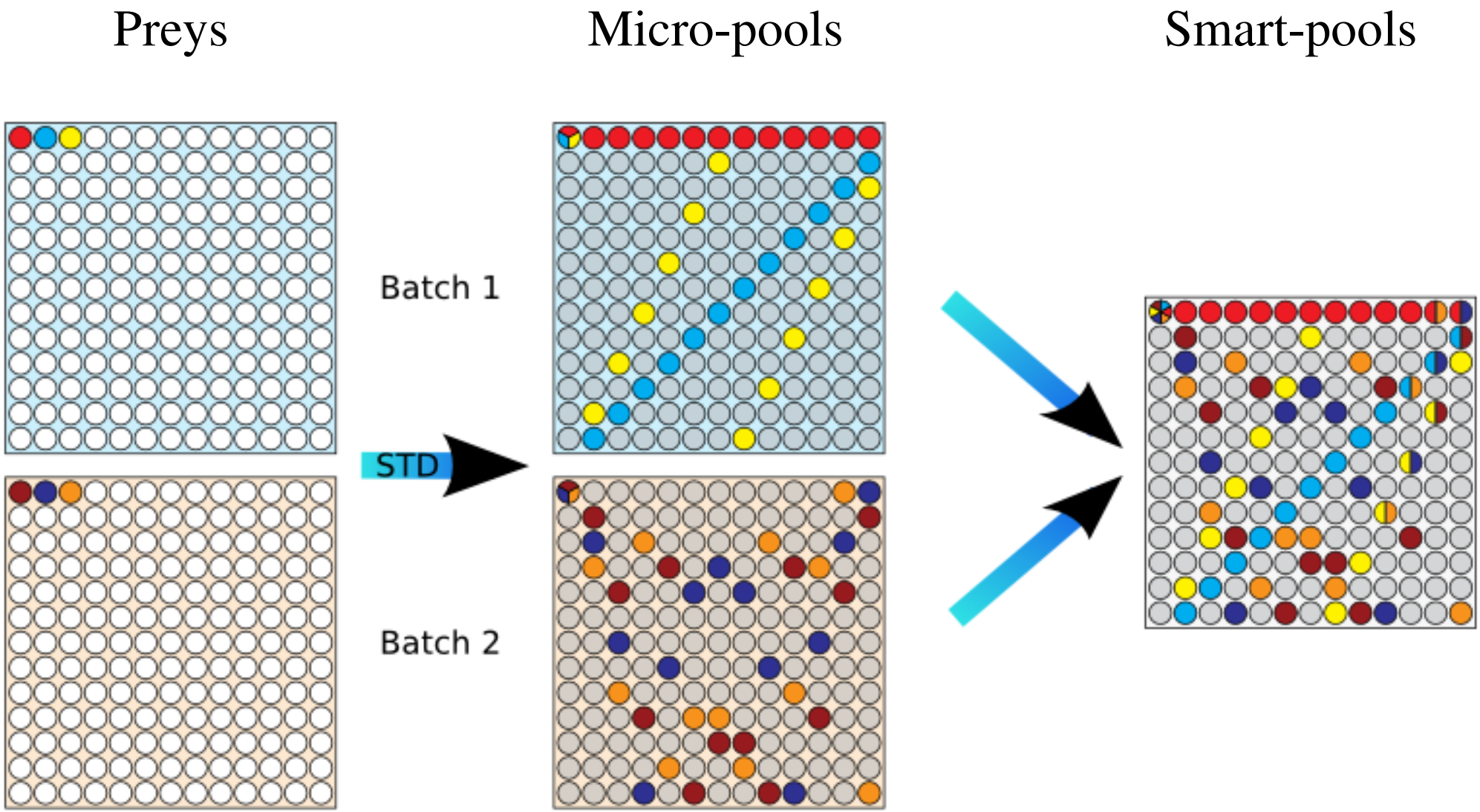
STD: small example



STD: small example




Preys
Micro-pools
Batch 1
STD
Batch 2


Preys
Micro-pools
Smart-pools
Batch 1
Batch 2
STD

STD Properties

- Theorem: number of pools that contain any 2 variables is at most $\Gamma(q,n)$

- Proof: layers j = roots of non-zero polynomial on GF(q) of degree at most Γ

- $\Gamma(q,n) = \min\{\gamma \mid q^{\gamma+1} \geq n\}$

  Example: n=940, q=13 => Γ=2

A solution to the pooling problem

- **Corollary:** If there are **at most t positive variables** in $\mathcal{A}_n$ and **at most E false positive and E false negative observations**:

STD(n;q;k) is a solution, when choosing q prime such that

$$t \cdot \Gamma(q,n)+2 \cdot E < q, \text{ and } k = t \cdot \Gamma+2 \cdot E+1$$

- Constructive proof: exhibit a simple algorithm that works

A solution to the pooling problem

- **Corollary:** If there are **at most t positive variables** in $\mathcal{A}_n$ and **at most E false positive and E false negative observations**:

STD(n;q;k) is a solution, when choosing q prime such that

$t \cdot \Gamma(q,n)+2 \cdot E < q$, and $k = t \cdot \Gamma+2 \cdot E+1$

- Constructive proof: exhibit a simple algorithm that works

  Algorithm relies on knowledge of E

- STD is sound
- Allows to compare with other designs: very favorable

(eg recently: D'yachkov *et al* 2005: 16x, Eppstein *et al* 2007: ~2x)

Even redistribution of variables

Theorem: Let $m \le k \le q$ and consider $\{P_1,\ldots,P_m\} \subset STD(n;q;k)$, each belonging to a different layer. Then:

$$\lambda_m \le \left|\bigcap_{h=1}^{m} P_h\right| \le \lambda_m + 1 \quad \text{, where} \quad \lambda_m = \sum_{c=m}^{\Gamma} \left(\left[\!\left[\frac{n-1}{q^c} \right]\!\right] \% q \right) q^{c-m}$$

- λ_m depends only on m, not on the choice of the pools $P_1,\ldots,P_m$

  $\rightarrow$ every pool, and every intersection between 2 or more pools, is redistributed evenly in each of the other layers

Using STD

- In practice: tolerate a few ambiguous variables → far fewer pools

  Example: n=10000, t=5, error-rate 1%

  - guarantee requires 483 pools
  - when tolerating up to 10 ambiguous variables, 143 pools prove sufficient

- Given (n,t,E-rates) and "ambiguity tolerance", find optimal parameter values (q,k) by simulation

- Difficulty: "decode" observed pool values

Talk layout

- The pooling problem: how to design pools?
- **The decoding problem: how to interpret results?**
- Experiments: Y2H interactome mapping
  - Human: 100 baits vs 940 preys
  - Worm: 12 baits vs 13000 preys

Interpreting smart-pooling results

Highly redundant designs can correct noise *in theory*

In practice, E unknown: “pooling problem algorithms” don't work

Decoding problem: given an observation, find the set of variables that minimizes the number of errors

The decoding problem

- Discrete test outcomes, eg: NONE, FAINT, WEAK, STRONG
- Observation: vector of outcome values of size D == number of pools
- Interpretation: boolean vector of size D
- Consistent interpretation: values can be picked for the variables s.t. the interpretation is obtained by OR-ing each pool's variables' values
- Canonical mapping of each outcome to {0,1}, eg NONE and FAINT are *a priori* negative, WEAK and STRONG are *a priori* positive
- Distance between each outcome and the other element of {0,1}, eg

$\delta_{NONE}=2,\ \delta_{FAINT}=1,\ \delta_{WEAK}=2,\ \delta_{STRONG}=4$

- Induced distance between observation and interpretation: sum over all pools
- The decoding problem: given an observation, find all consistent interpretations at minimal distance.

The decoding problem

Note: with 2 possible outcomes NEG and POS and $\delta_{NEG} = \delta_{POS} = 1$:

classical decoding problem implicit in CGT (distance = sum of errors)

Exponential search space

Example: 5 positives among 1000 preys => ~2^{50} consistent interpretations...

Combinatorial optimization problem

Difficult for general solvers (eg integer linear programming)

Interpool

- Interpool: an algorithm to solve it
  - Branch-and-bound
  - Exact
  - Fast (usually)
  - GNU GPL

Thierry-Mieg N, Bailly G. Bioinformatics. 2008 Mar 1;24(5):696-703.

Talk layout

- The pooling problem: how to design pools?
- The decoding problem: how to interpret results?
- **Experiments: Y2H interactome mapping**
  - **Human: 100 baits vs 940 preys**
  - Worm: 12 baits vs 13000 preys

Xin et al, Genome Research 2009

Validation vs CCSB-HI1

- 100 baits x 940 preys

  Varied subspace of CCSB-HI1: many interactions, hubs, auto-activators...

  Choosing the design: simulations with interpool

STD(940;13;13), 10% FPR

Positives	FNR	TPs missed	Retests	Simulations	Time
2	10%	0	2.26	10000	1m
	20%	0	2.26		1m
	30%	1.2%	2.27		4m
3	10%	0	3.57	10000	4m
	20%	0.4%	3.58		33m
	30%	3.4%	3.60		2h
4	10%	0	5.06	10000	32m
	20%	1.0%	5.11	10000	10h39m
	30%	6.2%	5.26	7500	2d11h
5	10%	0.1%	6.71	10000	4h
	20%	1.7%	6.94	1000	12h47m
	30%	12.9%	7.88	300	3d10h

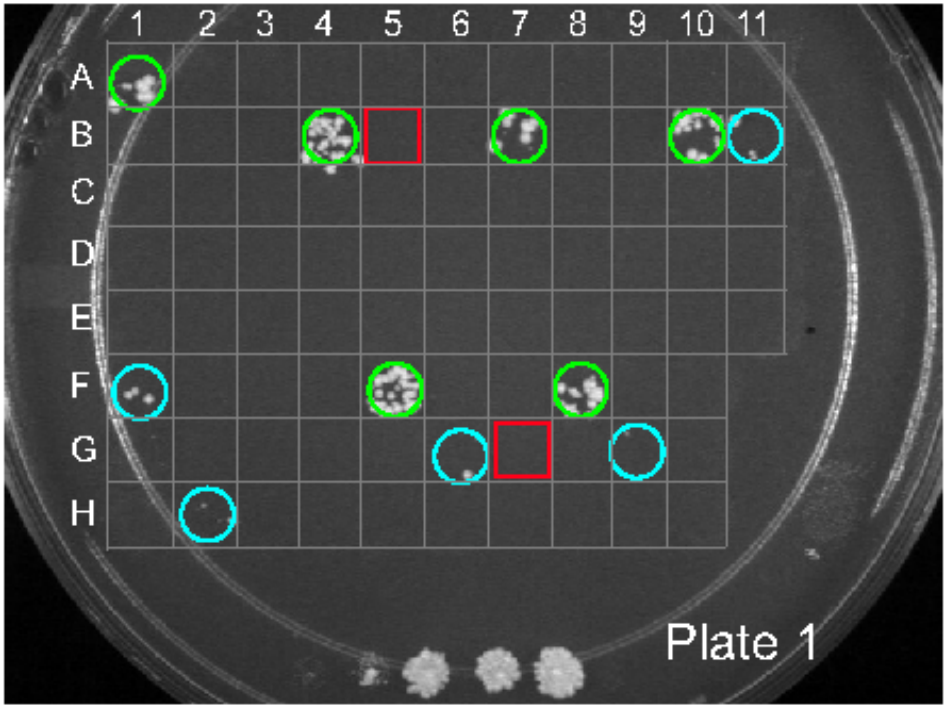
TPs missed & Retests: upper bounds of 95% confidence intervals

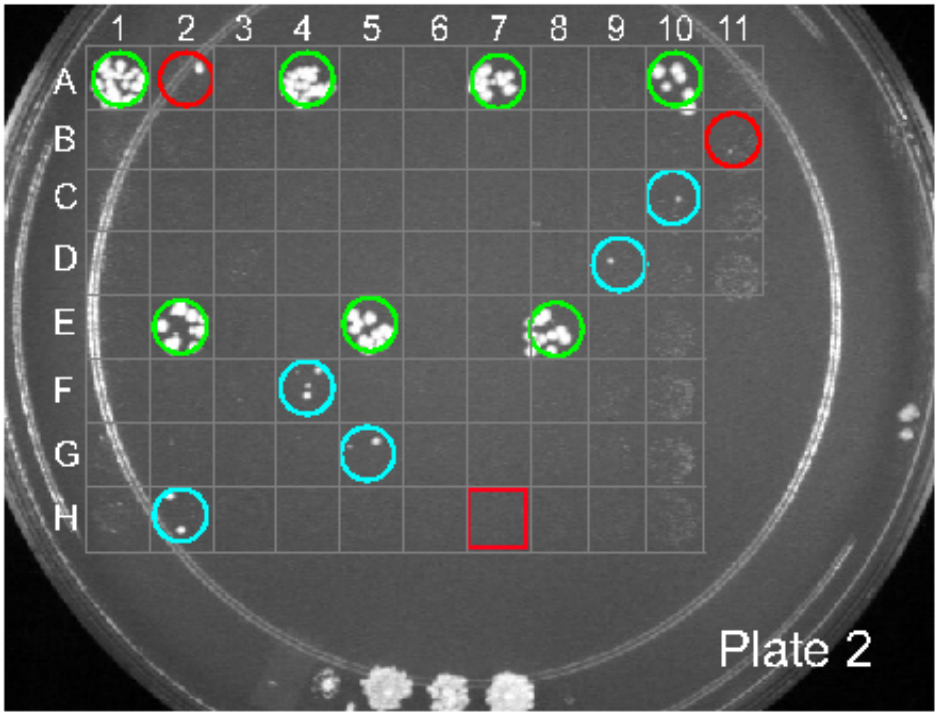
Validation vs CCSB-HI1

- Smart-pooled the 940 preys according to STD(940;13;13)
  - 169 pools, 73 preys in each pool
  - each prey is in 13 pools
  - at most 2 pools contain any pair

    → 3 pools for identification, 10 pools for errors and multiple positives (extra redundancy)
- Screened each bait against the 169 pools, scored positive pools
- Decoded (interpool) -> putative positives
- Pairwise retests

Example with one bait





Circles: spots scored positive.

Decoding finds:

- 2 interactors: **green** (no FNs), and **blue** (3 FNs = **red squares**)
- 2 FPs (**red circles**)

Results

- Identified 65 putative interactions
- Retest: 60 passed → **PPV = 92%**
- **Sensitivity vs CCSB-HI1**: between 1.5 and 2.8 times higher

Talk layout

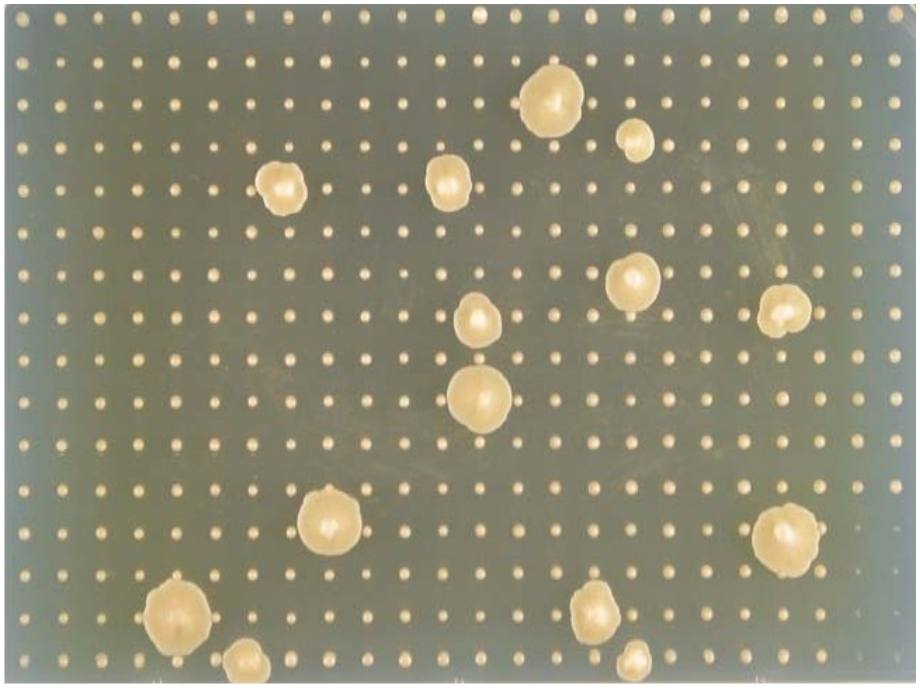
- The pooling problem: how to design pools?
- The decoding problem: how to interpret results?
- **Experiments: Y2H interactome mapping**
  - Human: 100 baits vs 940 preys
  - **Worm: 12 baits vs 13000 preys**

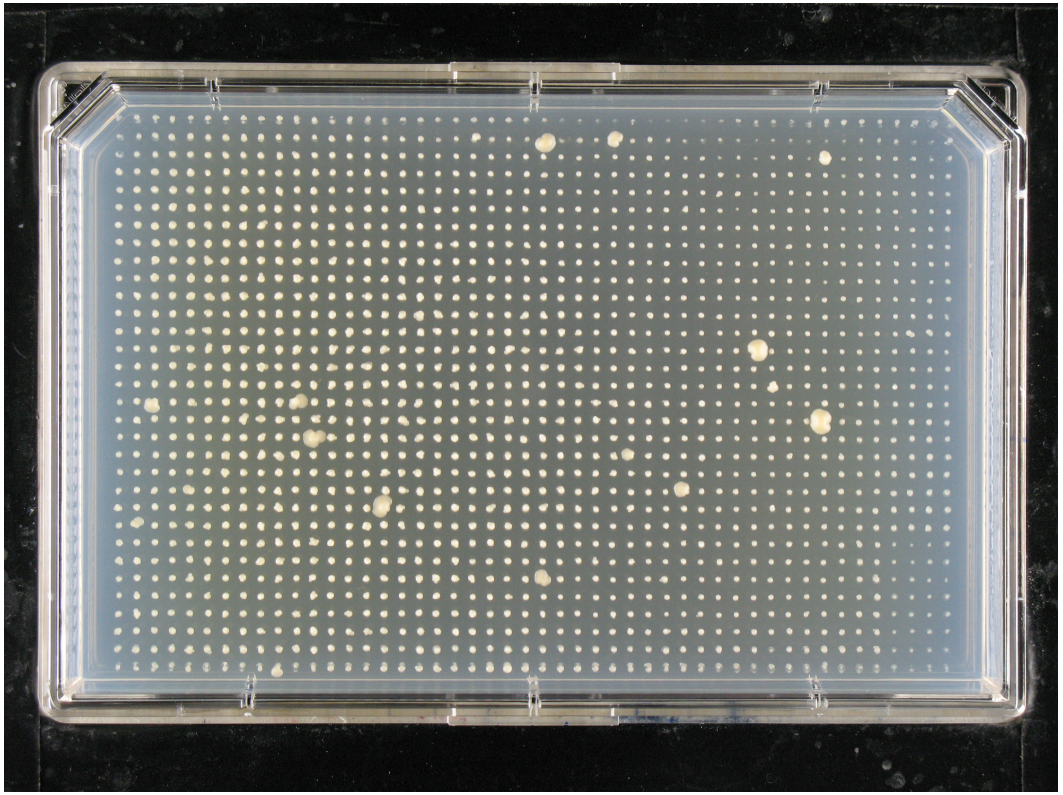
Xin et al, Genome Research 2009

ORFeome-wide worm screens

Scaling up to the complete *C. elegans* ORFeome, using denser formats (384 and 1536)

384 and 1536 formats





ORFeome-wide worm screens

Scaling up to the complete *C. elegans* ORFeome, using denser formats (384 and 1536)

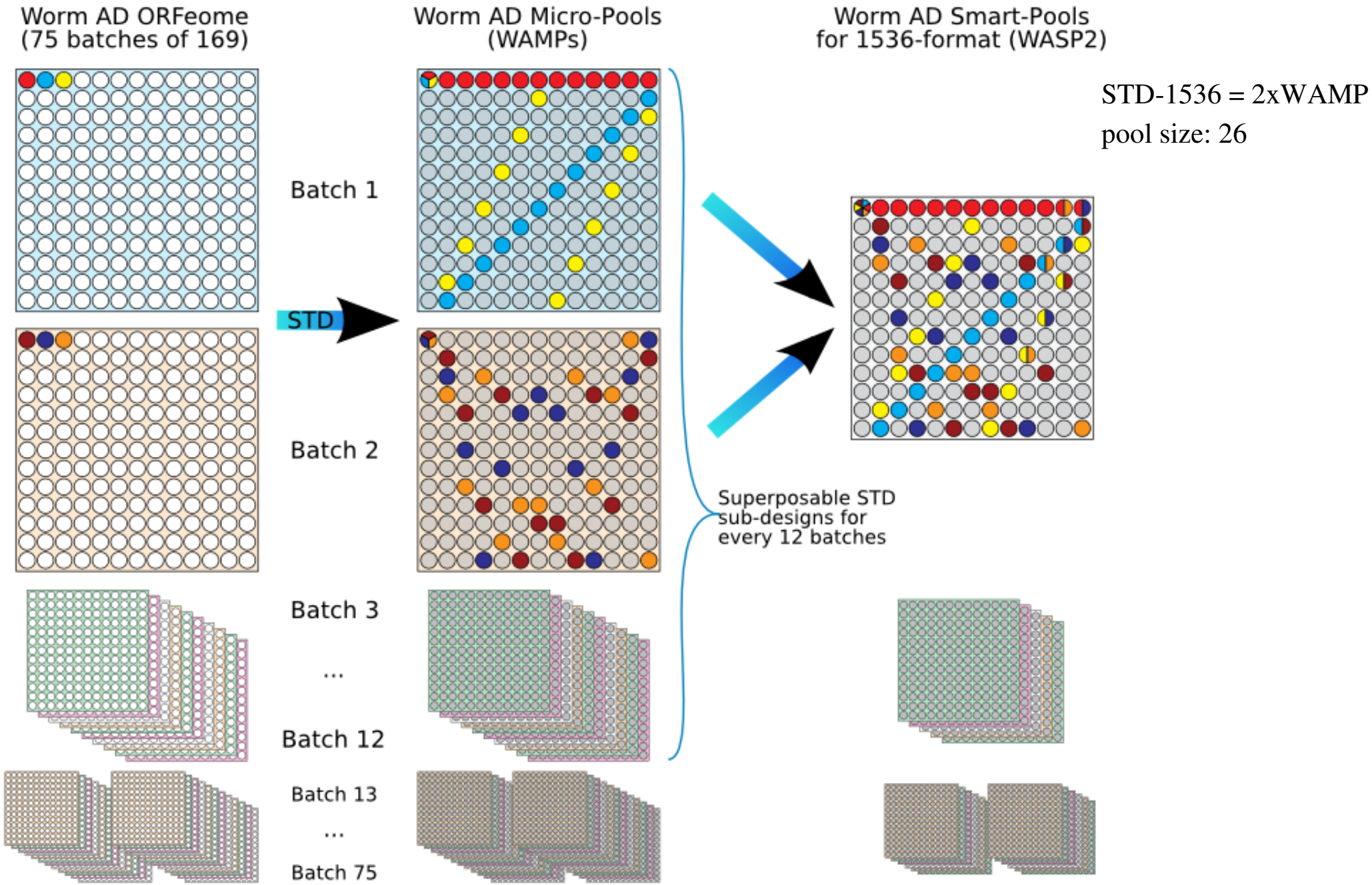
Screen 12 SH3 domains as baits with 4 methods:

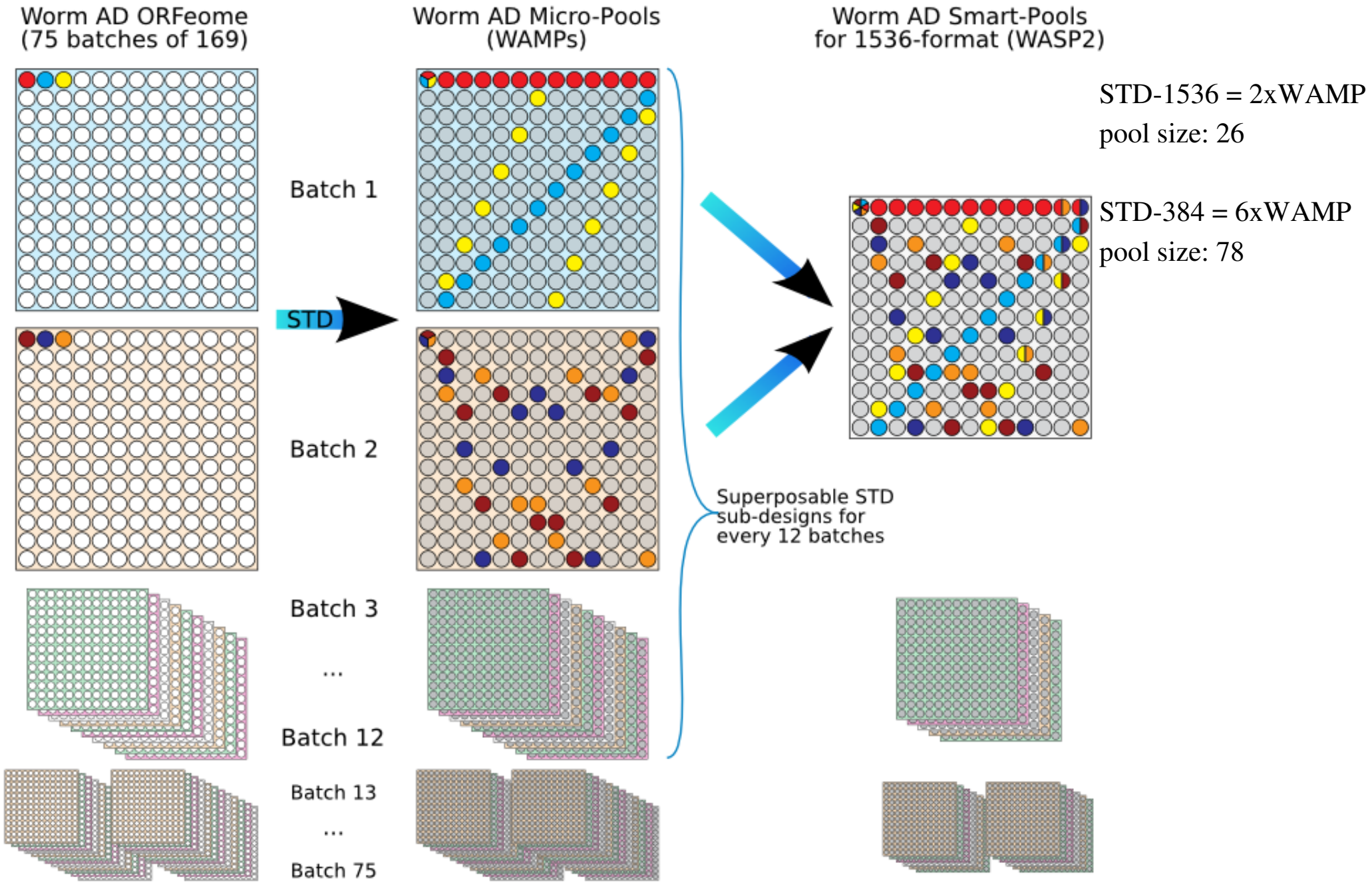
- 1-on-1: duplicated one-on-one Worm AD array
- Screen-Seq: colony-picking & sequencing (6 baits)
- STD smart-pooling designs: STD-384, STD-1536

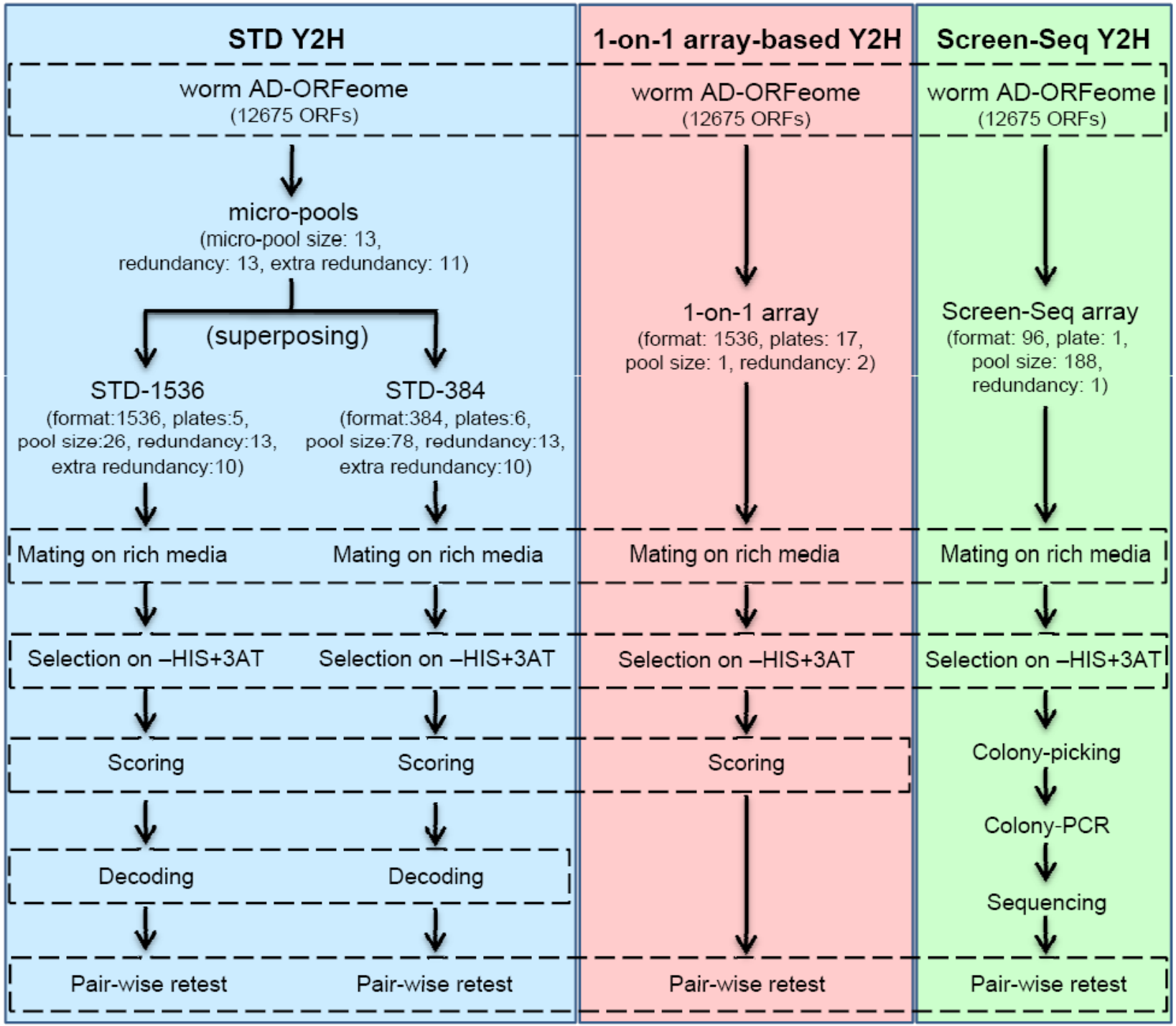
Every screen performed in duplicate

Pairwise retest of each hit in quadruplicate

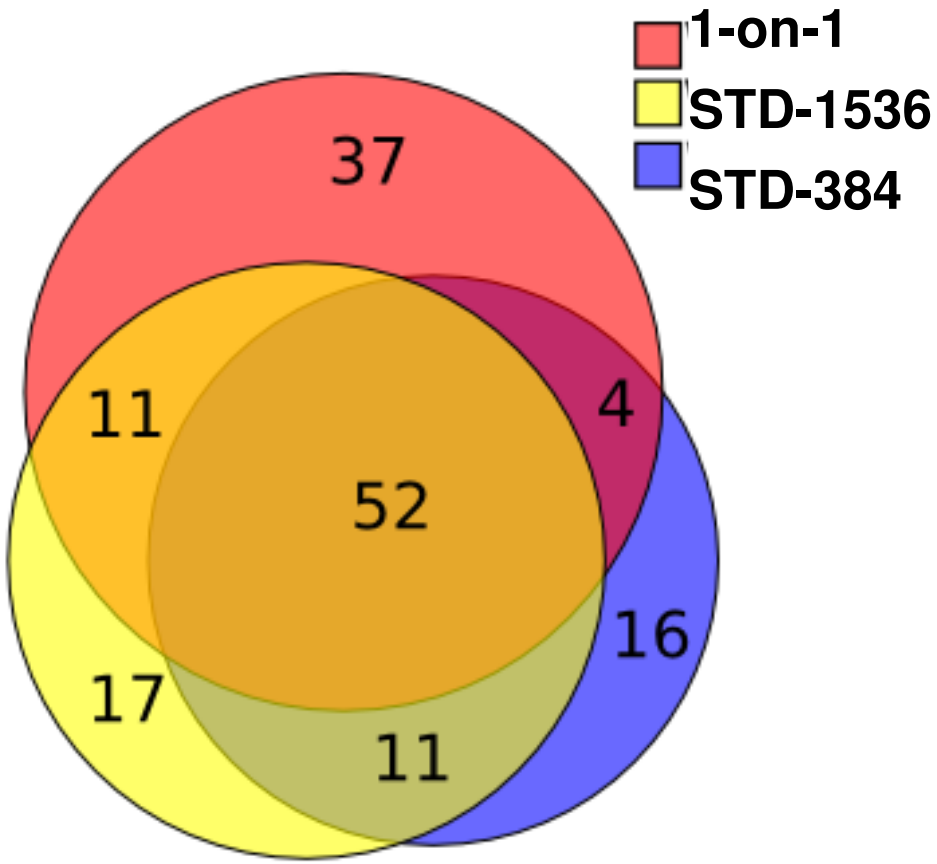
- Core (positives), NonCore (ambiguous), Neg (false positives)


Worm AD ORFeome
(75 batches of 169)
Worm AD Micro-Pools
(WAMPs)
Worm AD Smart-Pools
for 1536-format (WASP2)
STD-1536 = 2xWAMP
pool size: 26
Batch 1
STD
Batch 2
Batch 3
...
Batch 12
Batch 13
...
Batch 75
Superposable STD
sub-designs for
every 12 batches

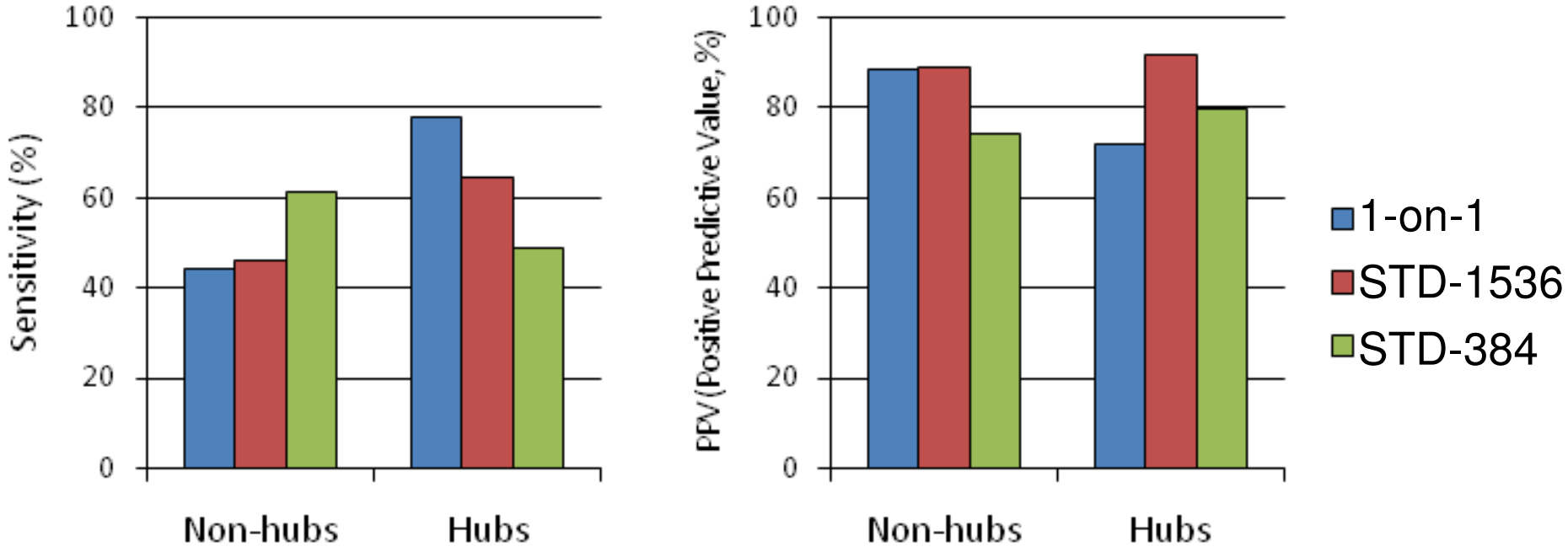

Worm AD ORFeome
(75 batches of 169)
Worm AD Micro-Pools
(WAMPs)
Worm AD Smart-Pools
for 1536-format (WASP2)
STD-1536 = 2xWAMP
pool size: 26
STD-384 = 6xWAMP
pool size: 78
Batch 1
STD
Batch 2
Batch 3
...
Batch 12
Batch 13
...
Batch 75
Superposable STD
sub-designs for
every 12 batches


STD Y2H
1-on-1 array-based Y2H
Screen-Seq Y2H
worm AD-ORFeome
(12675 ORFs)
worm AD-ORFeome
(12675 ORFs)
worm AD-ORFeome
(12675 ORFs)
micro-pools
(micro-pool size: 13,
redundancy: 13, extra redundancy: 11)
(superposing)
1-on-1 array
(format: 1536, plates: 17,
pool size: 1, redundancy: 2)
Screen-Seq array
(format: 96, plate: 1,
pool size: 188,
redundancy: 1)
STD-1536
(format:1536, plates:5,
pool size:26, redundancy:13,
extra redundancy:10)
STD-384
(format:384, plates:6,
pool size:78, redundancy:13,
extra redundancy:10)
Mating on rich media
Mating on rich media
Mating on rich media
Mating on rich media
Selection on –HIS+3AT
Selection on –HIS+3AT
Selection on –HIS+3AT
Selection on –HIS+3AT
Scoring
Scoring
Scoring
Colony-picking
Colony-PCR
Sequencing
Decoding
Decoding
Pair-wise retest
Pair-wise retest
Pair-wise retest
Pair-wise retest

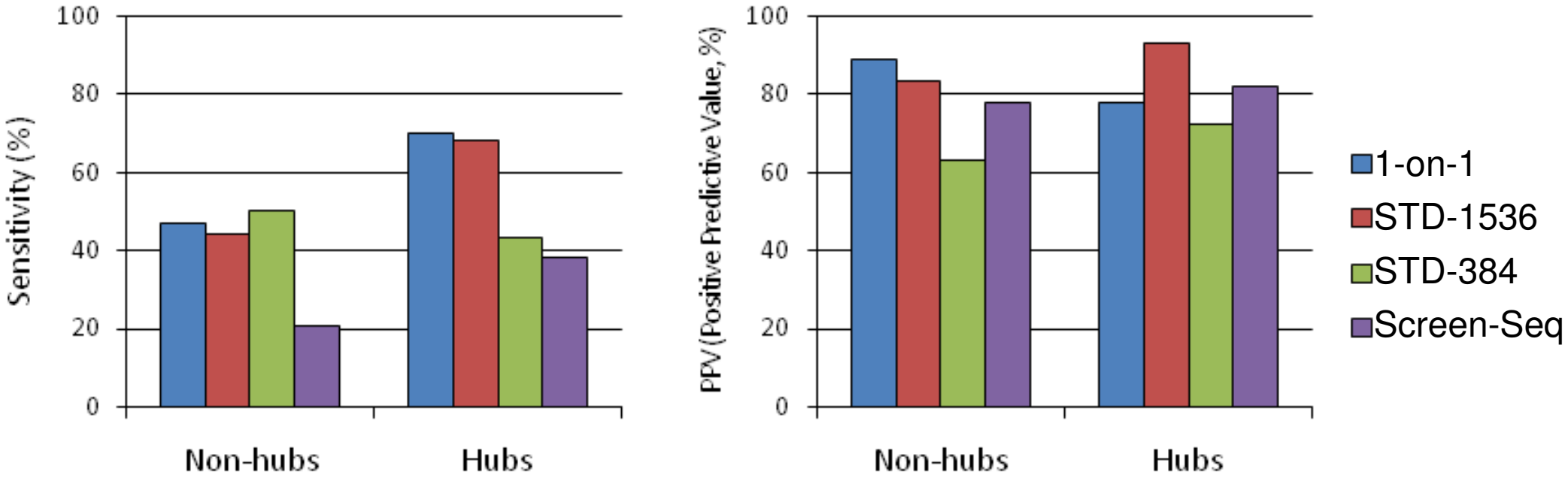
Worm results



STD vs 1-on-1

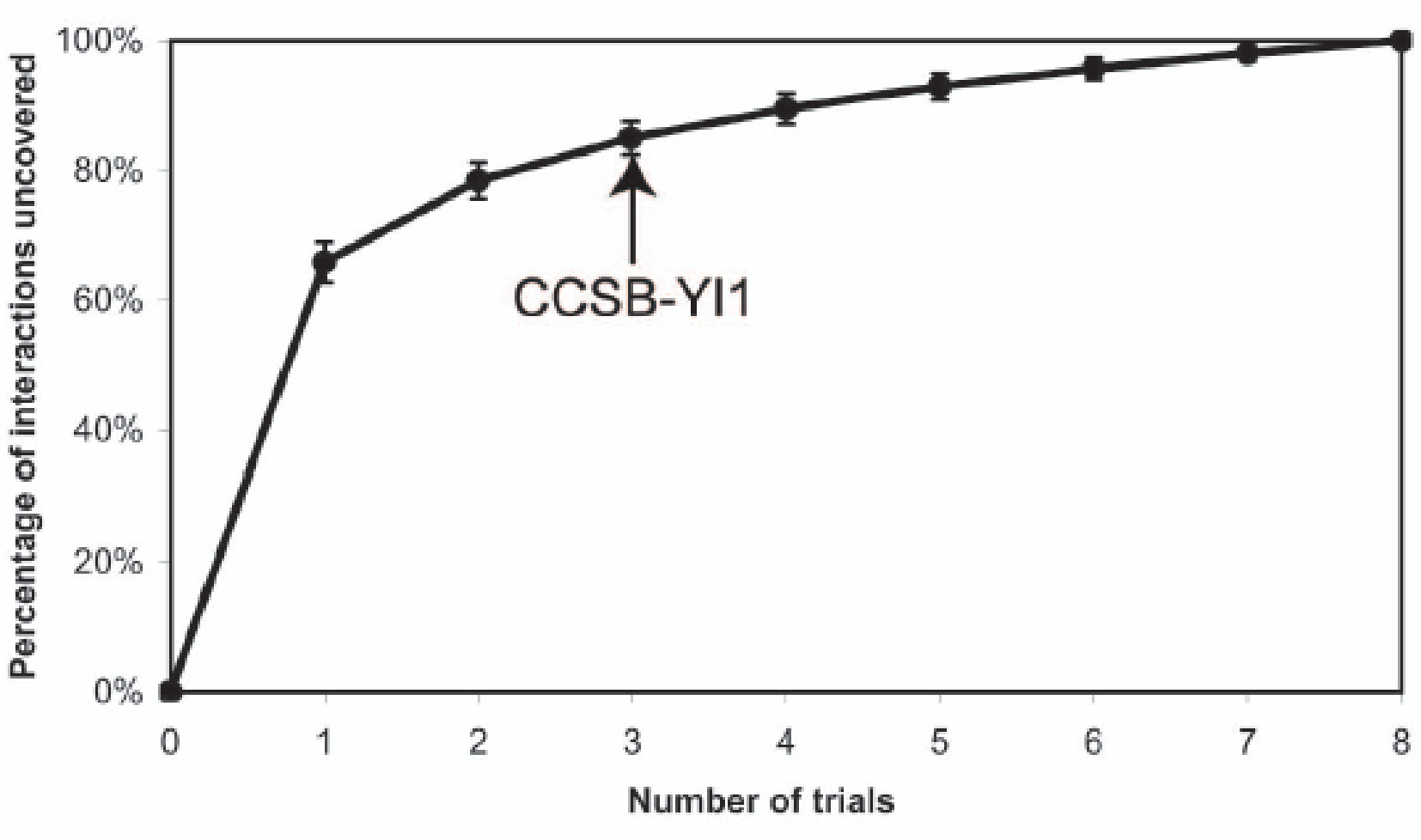

Sensitivity (%)
100
80
60
40
20
0
Non-hubs
Hubs
PPV (Positive Predictive Value, %)
100
80
60
40
20
0
Non-hubs
Hubs
1-on-1
STD-1536
STD-384

STD vs Screen-Seq



Worm screens: conclusions

- STD-based Y2H is highly sensitive and specific
- 1-on-1 is 3x more work & cost, no gain except for strong hubs
- Screen-Seq 1/3 less work & cost, but at least 2x less sensitive
  - Screen-Seq remains best for low-coverage maps

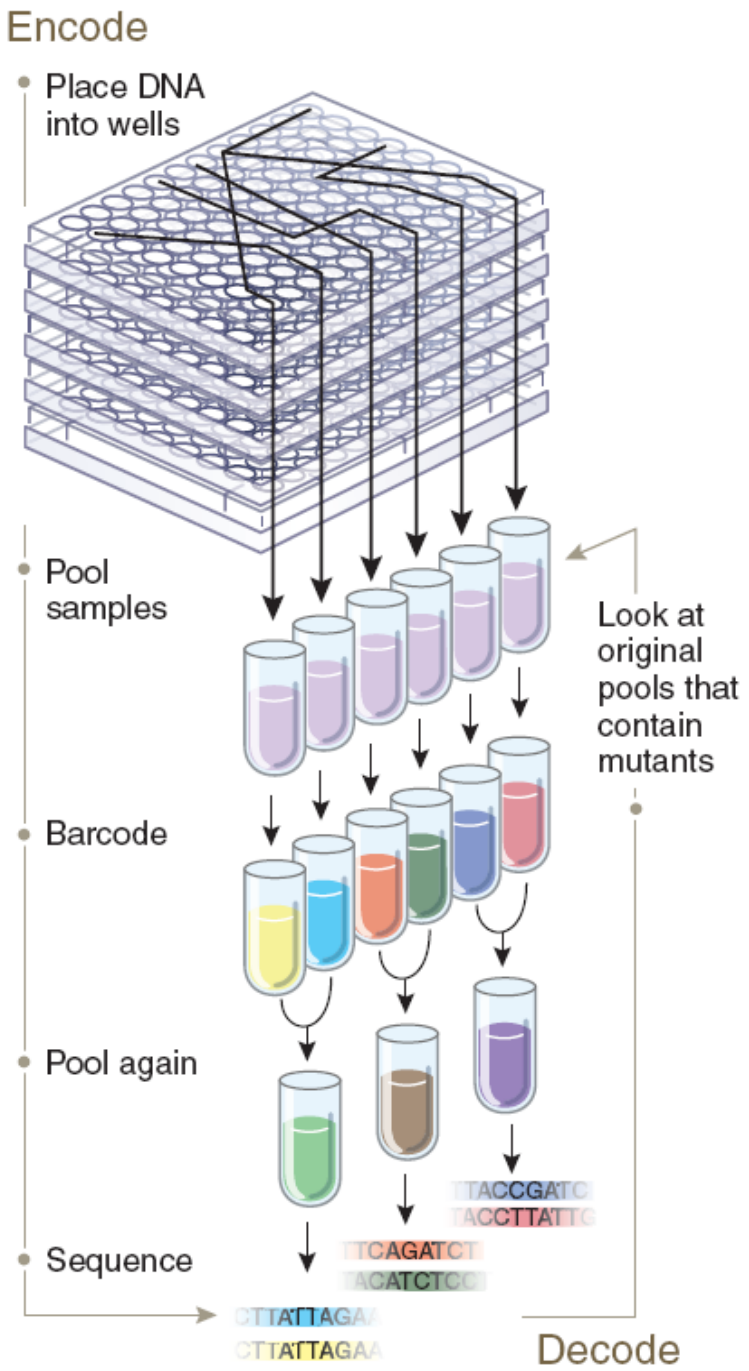


Yu et al, Science 2008

Worm screens: conclusions

- STD-based Y2H is highly sensitive and specific
- 1-on-1 is 3x more work & cost, no gain except for strong hubs
- Screen-Seq 1/3 less work & cost, but at least 2x less sensitive
  - Screen-Seq remains best for low-coverage maps
  - STD smart-pooling is much more efficient for high-coverage maps
- Could be useful for many assays

•Erlich et al, Genome Research 2009:
Identifying carriers of rare alleles in a collection of samples, using second-generation sequencing and barcoding.



Patterson & Gabriel,
Nat Biotech 2009

- Erlich et al, Genome Research 2009:
  Identifying carriers of rare alleles in a collection of samples, using second-generation sequencing and barcoding.

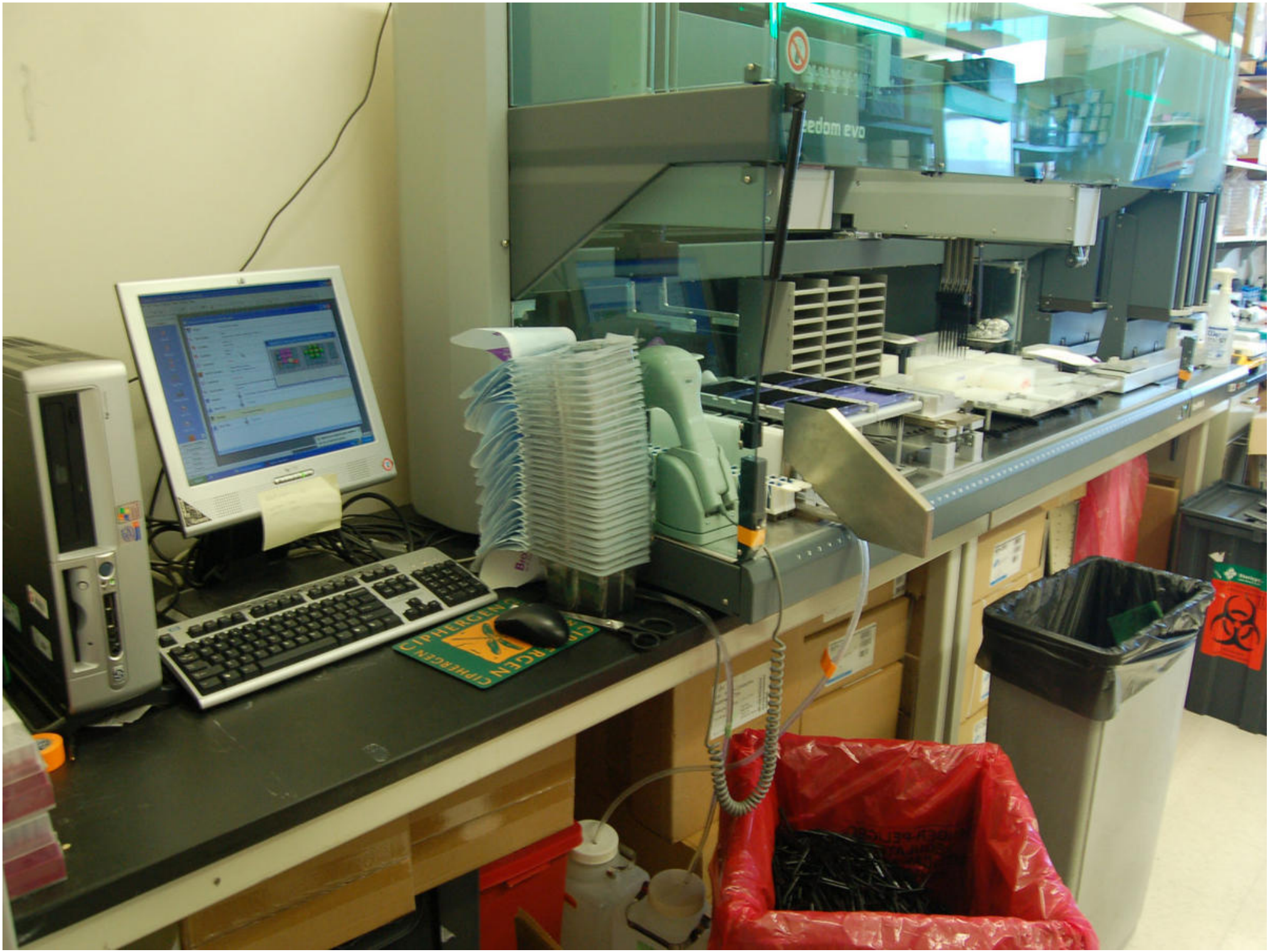
- Lonardi et al: *de novo* genome sequencing.

Summary

- **STD** (the Shifted Transversal Design) is a **flexible and efficient family of pooling designs**. On paper and in silico, STD performs very well.

- **Interpool** is a **fast exact decoding algorithm**. Useful both for choosing a design (simulations) and for interpreting experimental results. Open source.

- **STD-based smart-pooling really works** for HT-Y2H: it is efficient, sensitive and specific.


CIPHERGEN

Acknowledgments

M. Vidal
D. Hill
J.-F. Rual
T. Hirozane-Kishikawa

CCSB / DFCI, Boston

C. Boone
X. Xin

Boone Lab, University of Toronto

L. Trilling
G. Bailly

TIMC-IMAG and LIG, Grenoble