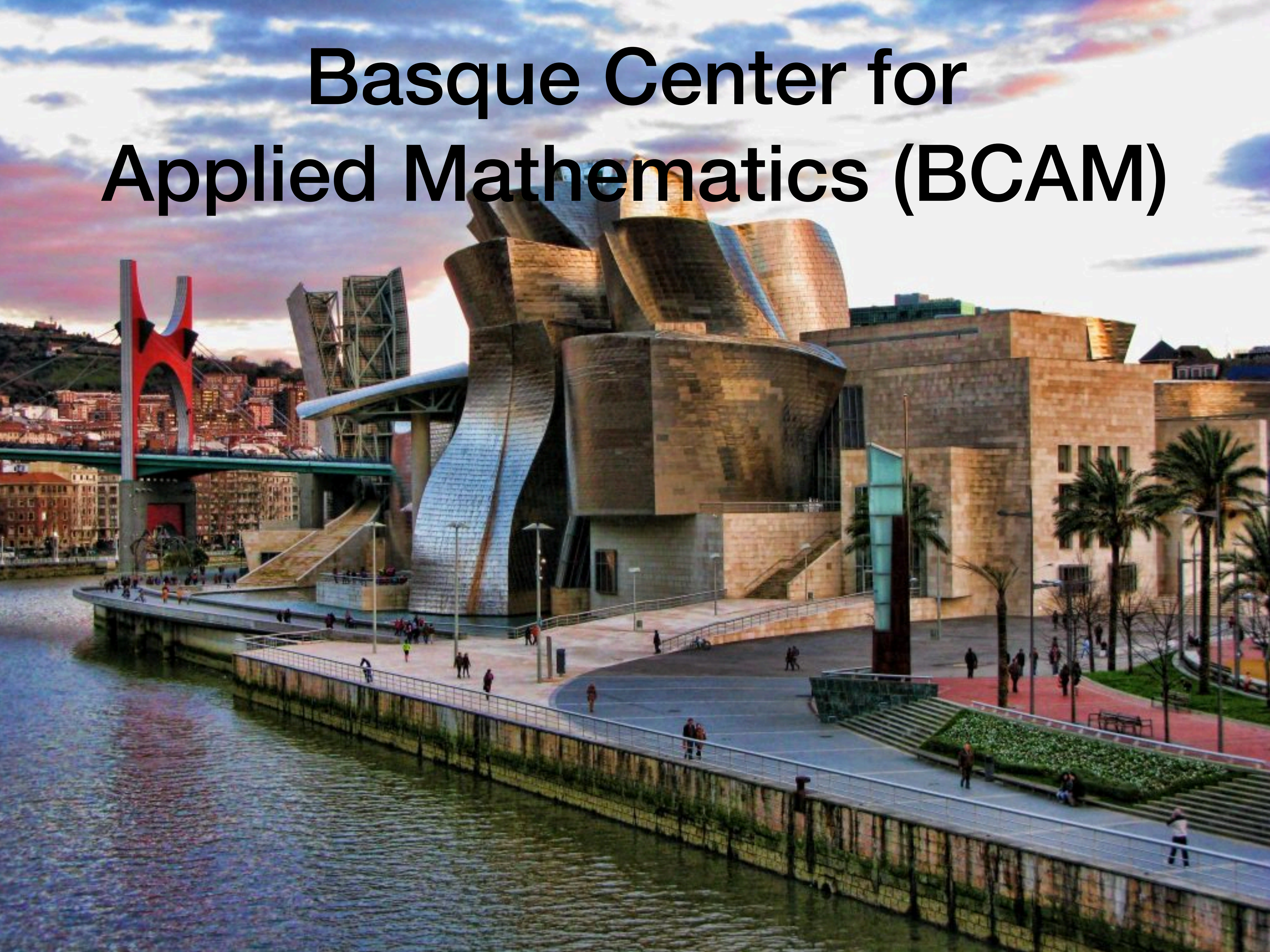
Probabilistic evolution and Permutation problems

Ekhine Irurozki

Basque Center for Applied Mathematics (BCAM)



Quiz

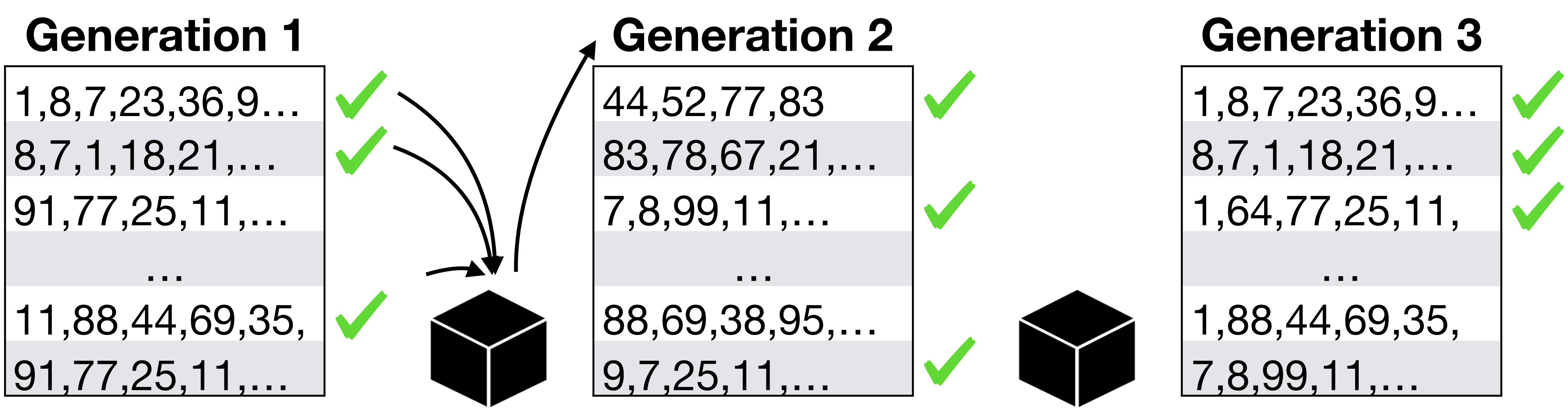
- Are you familiar with
  - Permutation problems
  - probabilistic EA
  - Statistics
  - Python / R

Outline

- Probability distributions
  - Sample and learn experimentally
  - EDA
- Permutation problems
  - Fitness functions and interpretations
- Probability distributions for permutations
  - EDA
- Distances for permutations

Evolutionary algorithms

- Roughly, evolution behaves like this

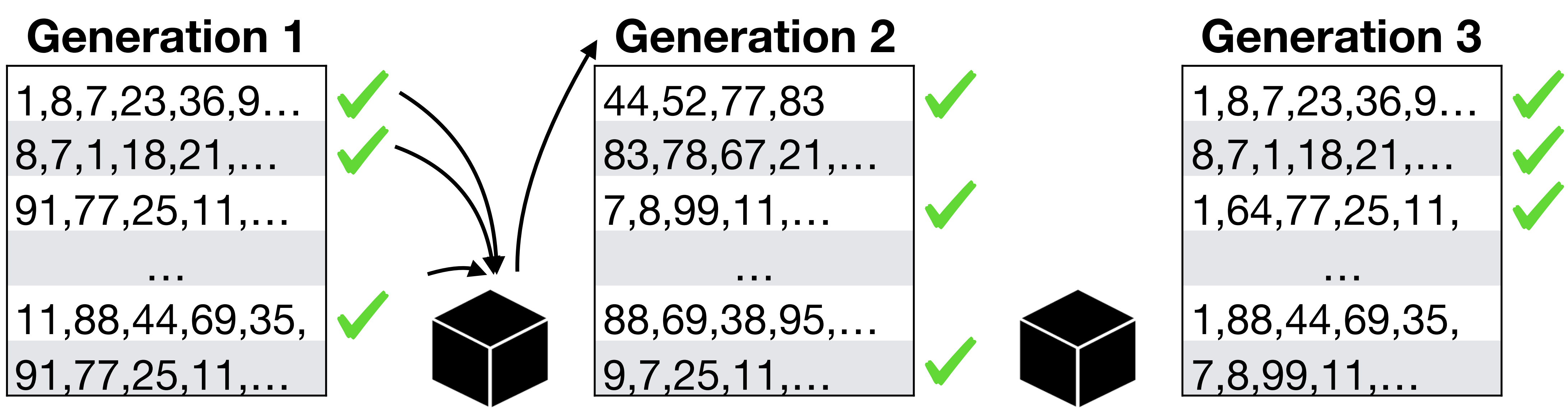


Crossover, mutation

- Combination of pairs of parents
  - Keep the best characteristics of both parents
- Sometimes is difficult to define crossover (we saw it yesterday)
- How do we get information about the evolution?

EDA

- Estimation of Distribution Algorithms
- Evolution is done by
  - p = learn the distribution of the best individuals in generation g
  - Generate a new generation $g+1$ by sampling p



Pros

- It can be used for every object for which a probability distribution can be defined

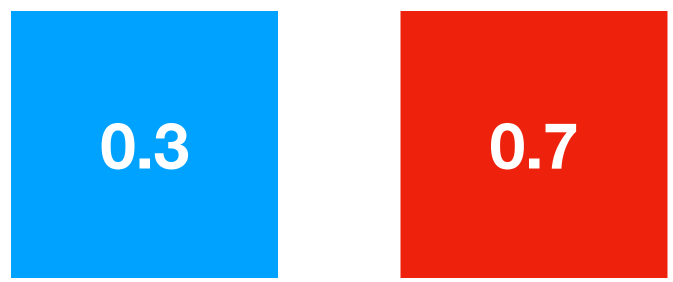
- At the end of the process you get a distribution over the final generation

Concepts

- Probability model a function that provides the probabilities of occurrence of different possible outcomes
- Random variable is a variable whose outcome depends on outcomes of a random phenomenon
- Sampling is the process of obtaining items that follow a given distribution
- Learning (estimating, parameter fitting) is the process of estimating the parameters of a probability distribution
  - *Estimation of distribution algorithms*

Bernoulli trial

- An event with two possible outcomes



- This is a Bernoulli trial of parameter 0.7

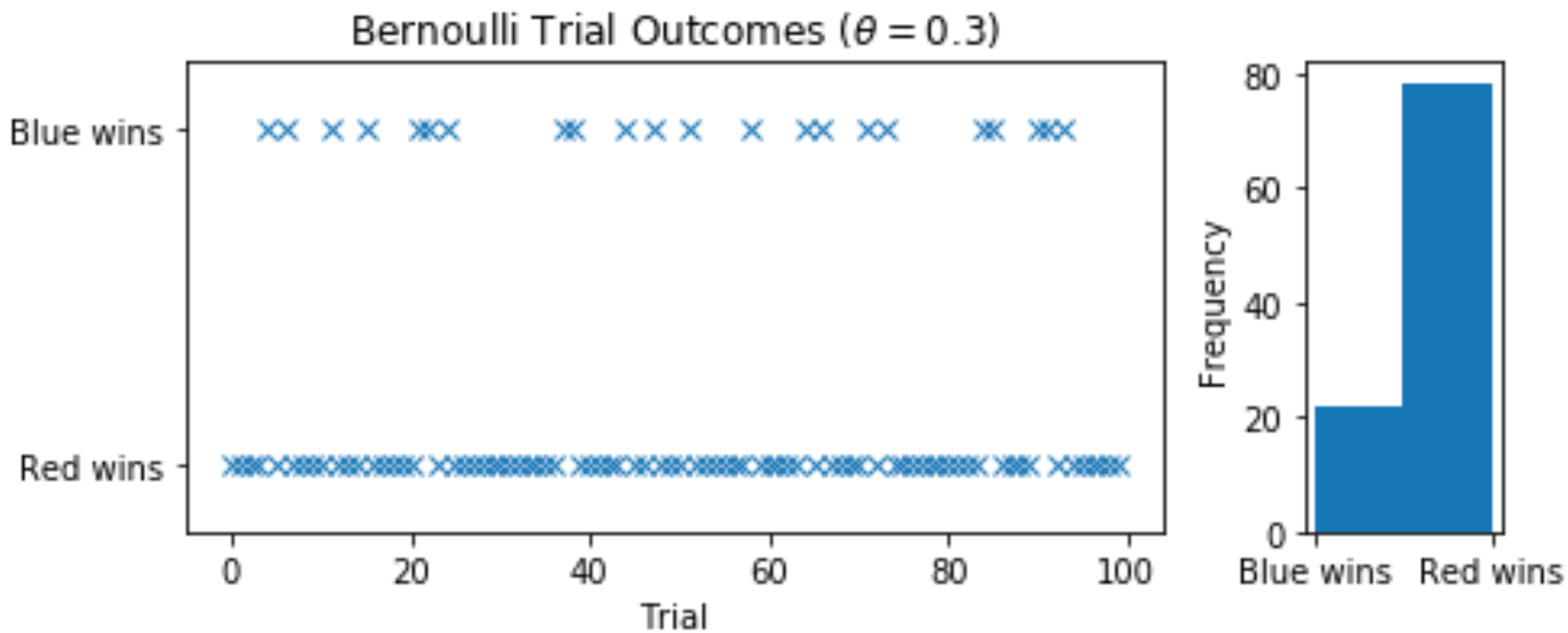
- The Binomial distribution models the distribution of the red team winning n times in p matches

Bernoulli

- $p(n, k, \theta) = \binom{n}{k} \theta^k (1 - \theta)^{n-k}$
- Sampling
  - If you have a random number generator [0,1]
  - In Python np.random.binomial

Exercise

- Replicate the experiment in the figure



Learning Bernoulli

- The estimator $\hat{p} = \dfrac{x}{n}$ is unbiased and has minimum variance

- We may not have enough samples. Then we use Laplace correction

Sampling Normal

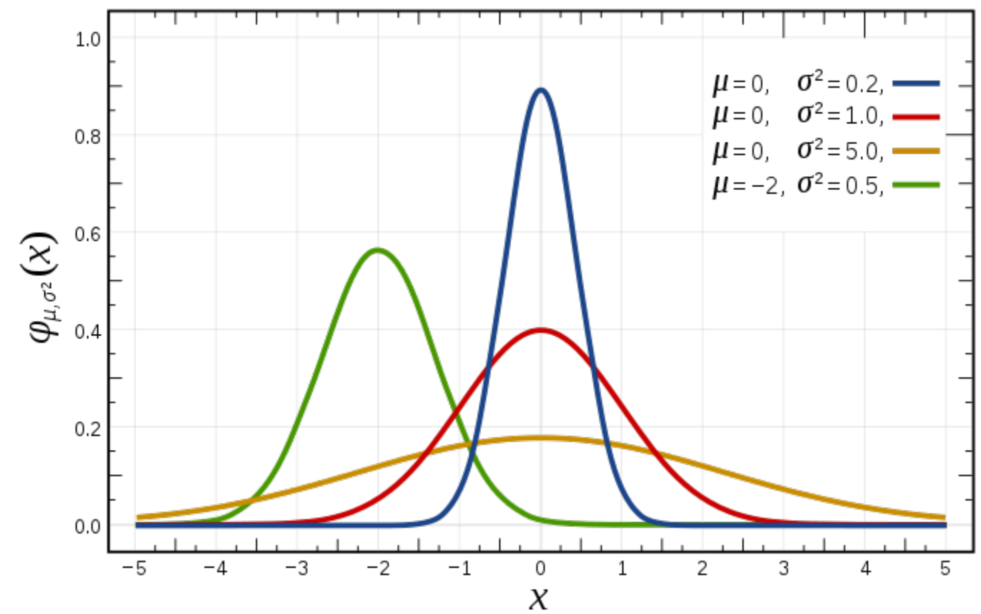
- $$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

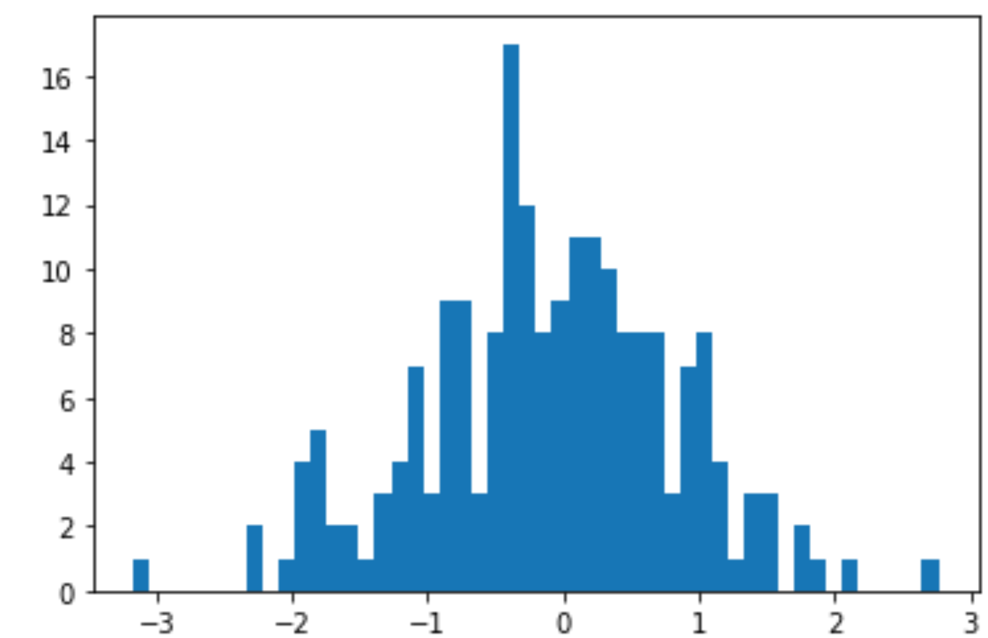
- We will sample with coding tools

- Exercise: simulate the experiment in the plot

  - rnorm(num, mean, std)

  - np.random.normal(mean, std, num)





Learning Normal

- We are given data points and want to know μ and σ
  - The empirical mean $\hat{\mu}$ is a *good* estimator of the real mean μ
  - The empirical variance is a *good* estimator of the real variance
    - *Good*??

Estimation quality

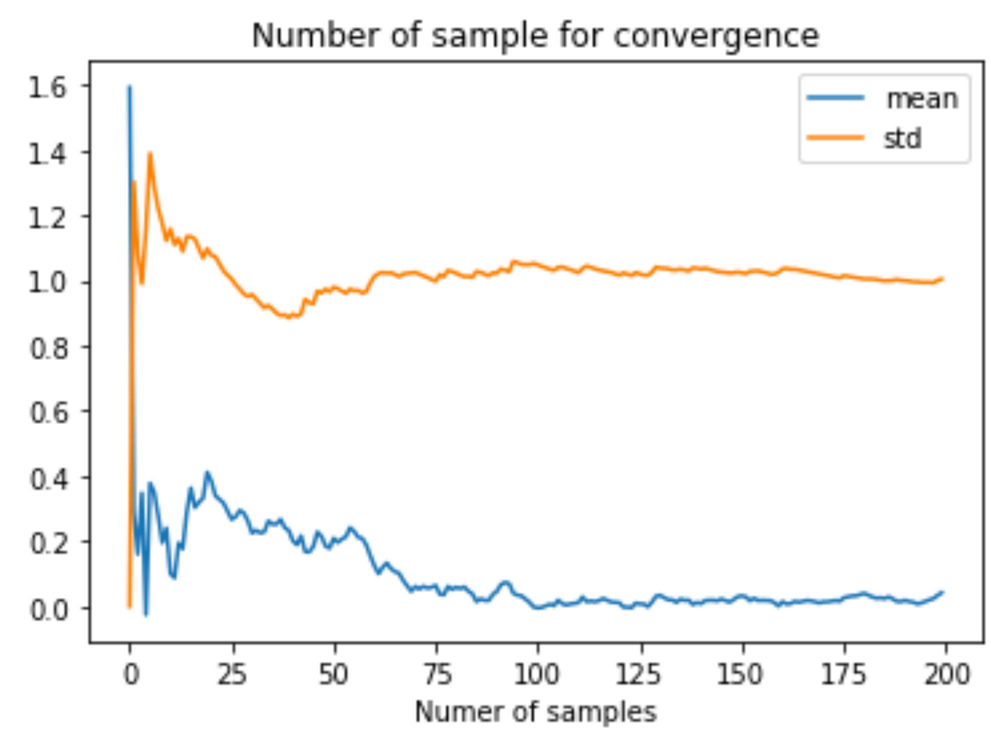
- Check some properties of the estimator: consistency, unbiasedness, confidence intervals for deviations, …
- How do we know that we are doing a good job empirically? sample and learn again
  - EA: How many samples do we need in our population?
  - ML: Do I need to to check all my data?

Exercise

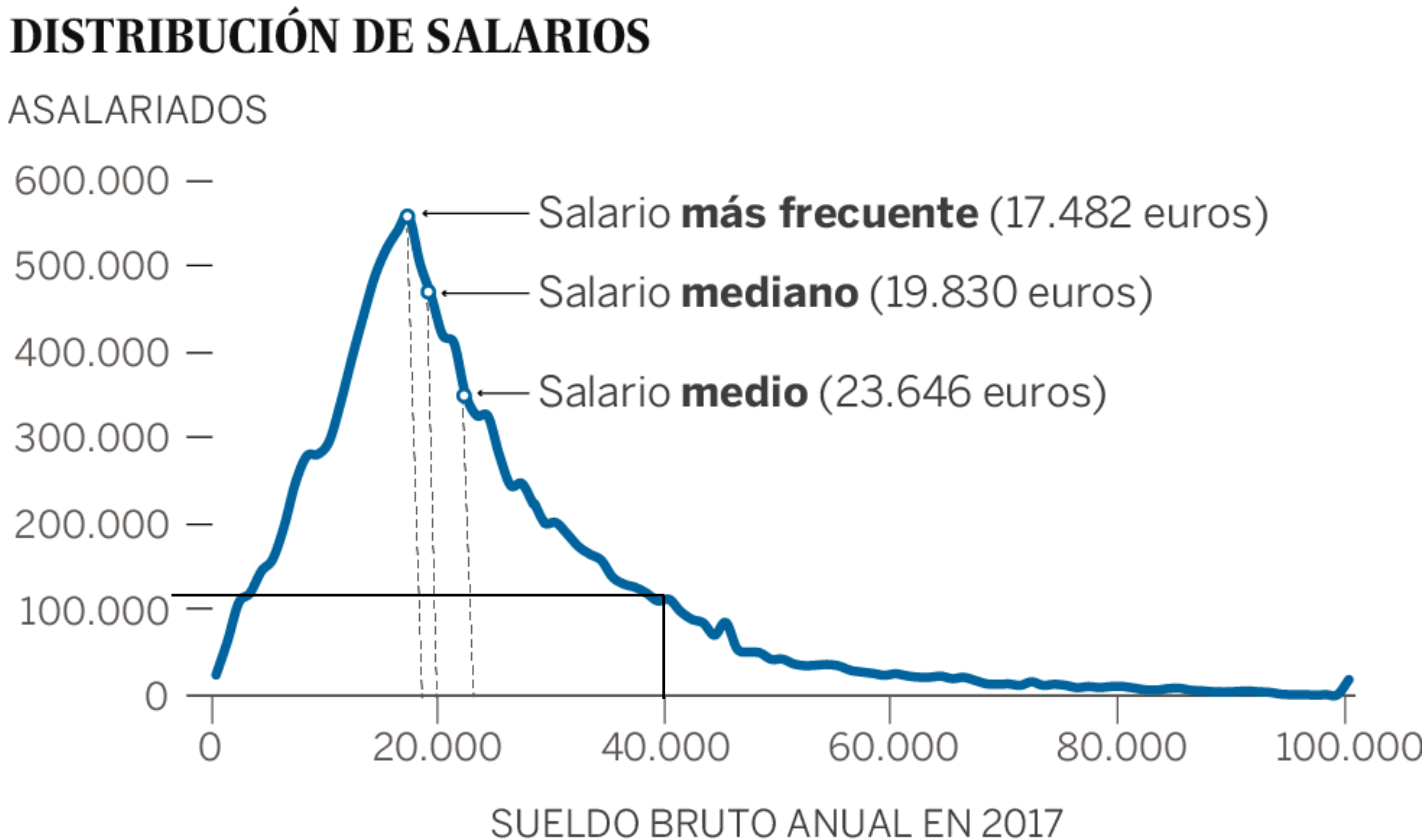
- Make a sample, learn the model and compare the estimates

- How many samples do we need for a correct estimation? This will define our population size

- Try this for different variances



Is the Normal always valid?



Is normal always valid?

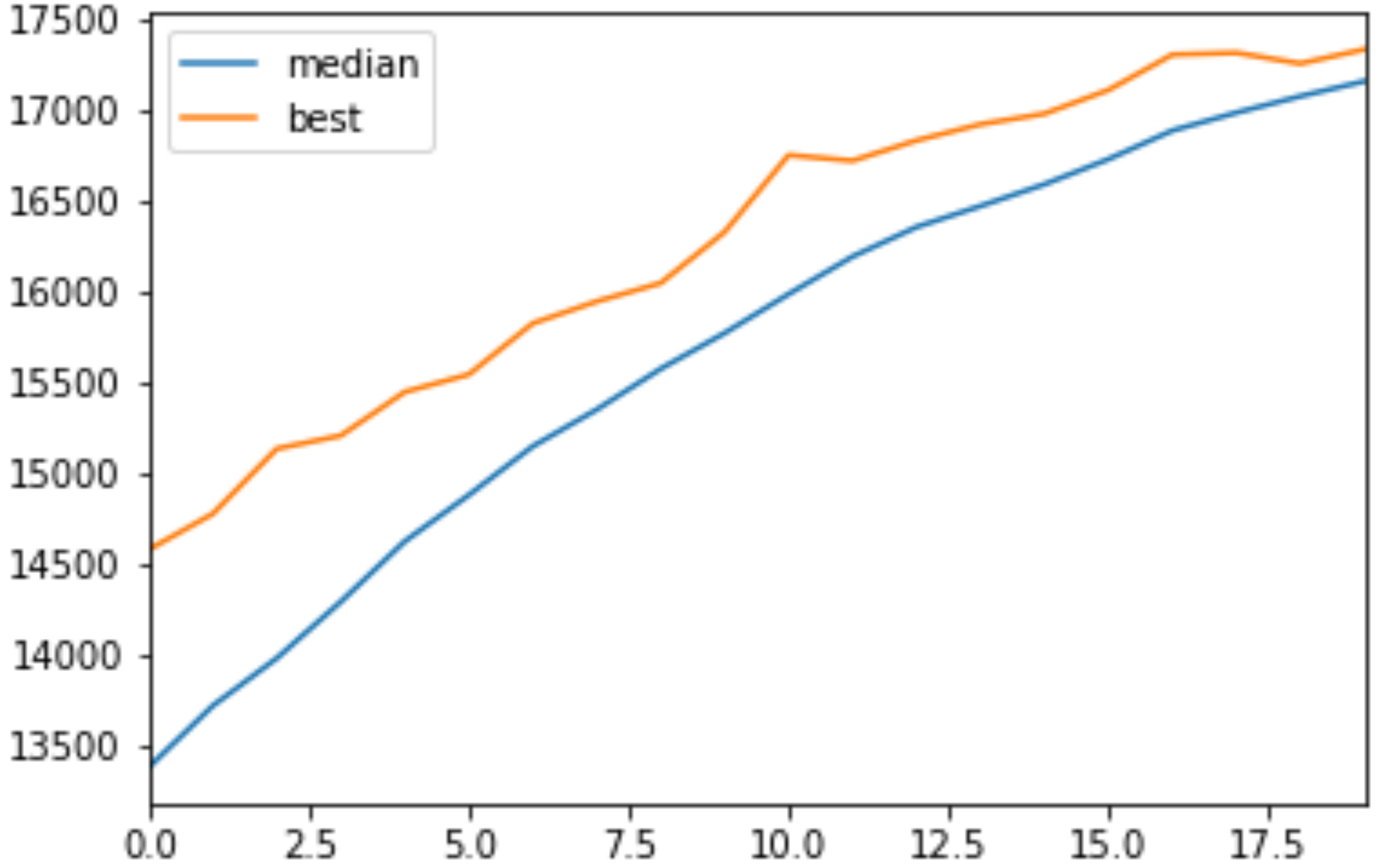
- Of course not but
  - It approximately holds on lots of real world processes
  - In some cases we just care on the variable in which the probability is maximum (the number of goals of a team)

UMDA

- Assume that the variables in our problem are independent
- MAXONE problem
  - A candidate solution is a string of n booleans
  - They are initialized randomly
  - Fitness is the number of ones in the string
  - What distribution do we learn and sample?

Exercise

- Code a UMDA



Permutation problems

Permutations

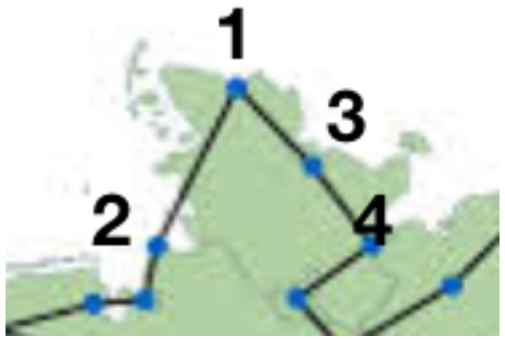
- Permutations are denote σ, π
- The number of permutations of n items is $n!$
- Grows faster than exponential, a^n
  - Can you prove it?

TSP

- Traveling salesman problem
- Cities in Germany is called 1, 2, 3, ..
- We have the distances between all the pairs of cities
- The goal is to the route along all the cities (once) with the shortest total distance

TSP

- Codification of a solution as a permutation
    - $\sigma = \ldots, 2, 1, 3, 4, \ldots$



- The fitness function is the sum of the distances
- Exercise
    - Given σ and D, how do you formalize the cost of the route?
    - Given σ how can we compute $\sigma\tau$?

TSP

- The trivial approach requires evaluating all the permutations
    - Which are the obvious improvements?
- The most elaborated approach is still $O(2^n)$
- The Germany instance with 15112 towns requires more than 22 years in a single 500MHz Alpha processor
- Applications in Routing, Biology, …

Crossover for routes

- Single point crossover in a rout does not make sense
- 0,1,2,3,4,5,6,7,8,9
- 4,3,1,7,0,9,6,5,8,2

LOP

- Linear ordering problem
- Find a permutation σ of rows and columns such that the sum of elements above the main diagonal is maximized
- Exercise, formalize the problem statement for matrix M

LOP

	1	2	3	4	5
1	0	16	11	15	7
2	21	0	14	15	9
3	26	23	0	26	12
4	22	22	11	0	13
5	30	28	25	24	0

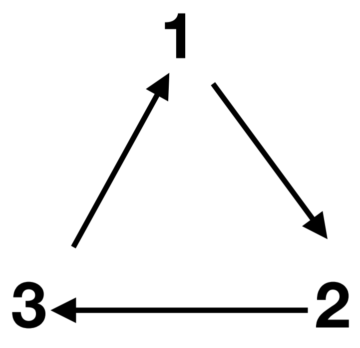
(a) $e = (1, 2, 3, 4, 5)$
$f(e) = 138.$

	2	3	1	4	5
2	0	14	21	15	9
3	23	0	26	26	12
1	16	11	0	15	7
4	22	11	22	0	13
5	28	25	30	24	0

(b) $\sigma = (2, 3, 1, 4, 5)$
$f(\sigma) = 158.$

Feedback arc set

- Matches between pairs of teams in a league
  - Team 1 beats team 2
  - Team 2 beats team 3
  - Team 3 beats team 1
- How do we rank the teams? Which is the winner?



	1	2	3	4
1	0	1	0	…
2	0	0	1	…
3	1	0	0	…
	…	…	…	0

FAS

- Which edges do we remove so that there is no cycle in the graph?

	1	2	3	4
1	0	4	1	…
2	8	0	3	…
3	2	6	0	…
	…	…	…	0

Kemeny ranking

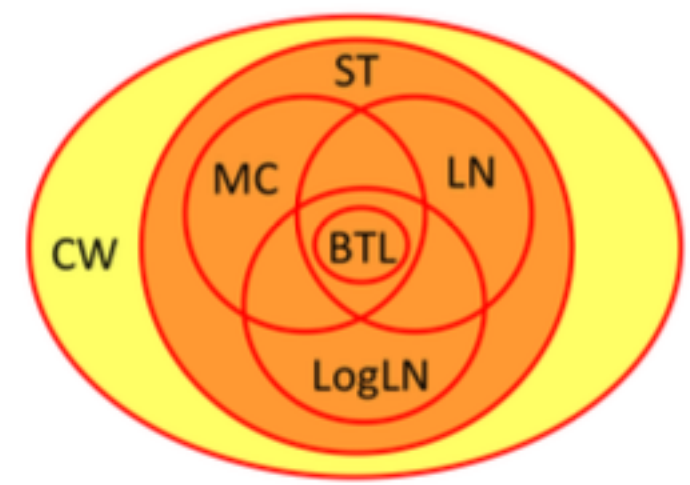
```
[1 0 4 2 3]
[4 1 2 3 0]
[3 1 4 0 2]
[0 3 1 2 4]
[4 0 1 2 3]
[1 0 4 3 2]
[0 4 2 1 3]
[2 4 3 1 0]
[3 0 4 1 2]
[3 2 0 1 4]
```

- Given a sample of permutations find the permutation that minimizes the sum of the distances to those in the sample. Formalize it
- Define $M_{i,j} = \frac{1}{m} \sum_{\sigma \in S} 1[\sigma(i) > \sigma(j)]$

```
[[0. , 0.7, 0.3, 0.6, 0.5],
 [0.3, 0. , 0.4, 0.5, 0.3],
 [0.7, 0.6, 0. , 0.6, 0.6],
 [0.4, 0.5, 0.4, 0. , 0.3],
 [0.5, 0.7, 0.4, 0.7, 0. ]]
```

Kemeny ranking

Condition on P	Property satisfied by P
Bradley-Terry-Luce (BTL)	$\exists \mathbf{w} \in \mathbb{R}^n_{++} : P_{ij} = \frac{w_i}{w_i + w_j}$
Low-noise (LN)	$P_{ij} > \frac{1}{2} \Rightarrow \sum_k P_{ik} > \sum_k P_{jk}$
Logarithmic LN (LogLN)	$P_{ij} > \frac{1}{2} \Rightarrow \sum_k \ln(\frac{P_{ik}}{P_{ki}}) > \sum_k \ln(\frac{P_{jk}}{P_{kj}})$
Markov consistency (MC)	$P_{ij} > \frac{1}{2} \Rightarrow \pi_i > \pi_j$
Stochastic transitivity (ST)	$P_{ij} > \frac{1}{2}, P_{jk} > \frac{1}{2} \Rightarrow P_{ik} > \frac{1}{2}$
Condorcet winner (CW)	$\exists i : P_{ij} > \frac{1}{2} \; \forall j \neq i$
Noisy permutation (NP)	$\exists \sigma \in \mathcal{S}_n, p < \frac{1}{2} : P_{ij} = \begin{cases} 1-p & \text{if } i \succ_\sigma j \\ p & \text{otherwise} \end{cases}$
Low rank (LR(ψ, r))	$\mathrm{rank}(\psi(\mathbf{P})) \leq r \;\; (\psi : [0,1] \to \mathbb{R}, r \in [n])$



Ranking Algorithm	Finds optimal ranking?				
	BTL	LN	LogLN	MC	ST
Matrix Borda	✓	✓	×	×	×
BTL-MLE	✓	✓	×	×	×
Least Squares Ranking	✓	×	✓	×	×
Rank Centrality	✓	×	×	✓	×
SVM-Based Ranking	✓	✓	✓	✓	✓
Topological Sort	✓	✓	✓	✓	✓
Matrix Copeland	✓	✓	✓	✓	✓

- The three problems are equivalent

Assignment problems

- There are n agents and m tasks. The cost of assigning task i agent j is p_{ij}. Assign one task to each agent in such a way that the total cost is minimized.

- Hungarian algorithm gets a polynomial solution for balanced problems

- When the cost of all agents is the same as the cost of all tasks then we have the linear assignment problem LAP

- In general, we consider doubly stochastic cost matrices

QAP

- There are n locations and n facilities. The distance between facilities is given by matrix D and the flow between facilities by F. Assign facilities to locations in such a way that the product of distances and flows is minimized

Before EDAs

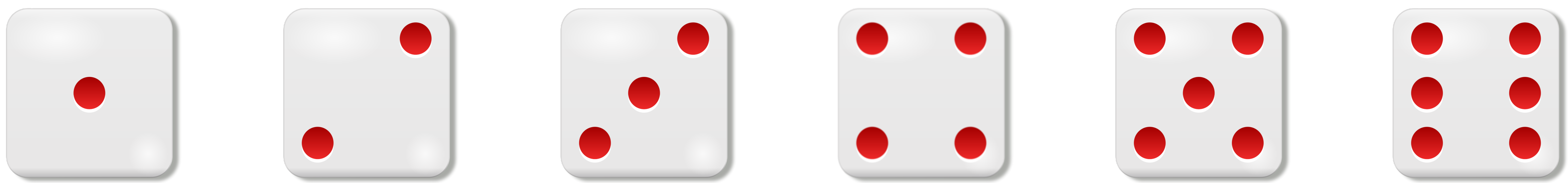
- Several problems
    - TSP
    - LOP ≡ FAS ≡ Kemeny ranking
    - LAP and QAP
- Candidate solutions as permutation, interpretation
- Fitness function
- So far everything is valid for any evolutionary algorithm

EDAs for permutation problems

- Evolutionary algorithms with a distribution over the objects of interest
  - Distribution for permutations with efficient sampling and learning operations
- UMDA is based on the independence of the variables, how do we translate this?
- What is a distribution for permutations?

Discrete distributions

- Probabilities in a fair dice



1/6	**1/6**	**1/6**	**1/6**	**1/6**	**1/6**

- Probabilities in a non fair dice

2/6	**0.5/6**	**0.5/6**	**1/6**	**1/6**	**1/6**

  - A distribution is a function that assigns a probability values to each possibility

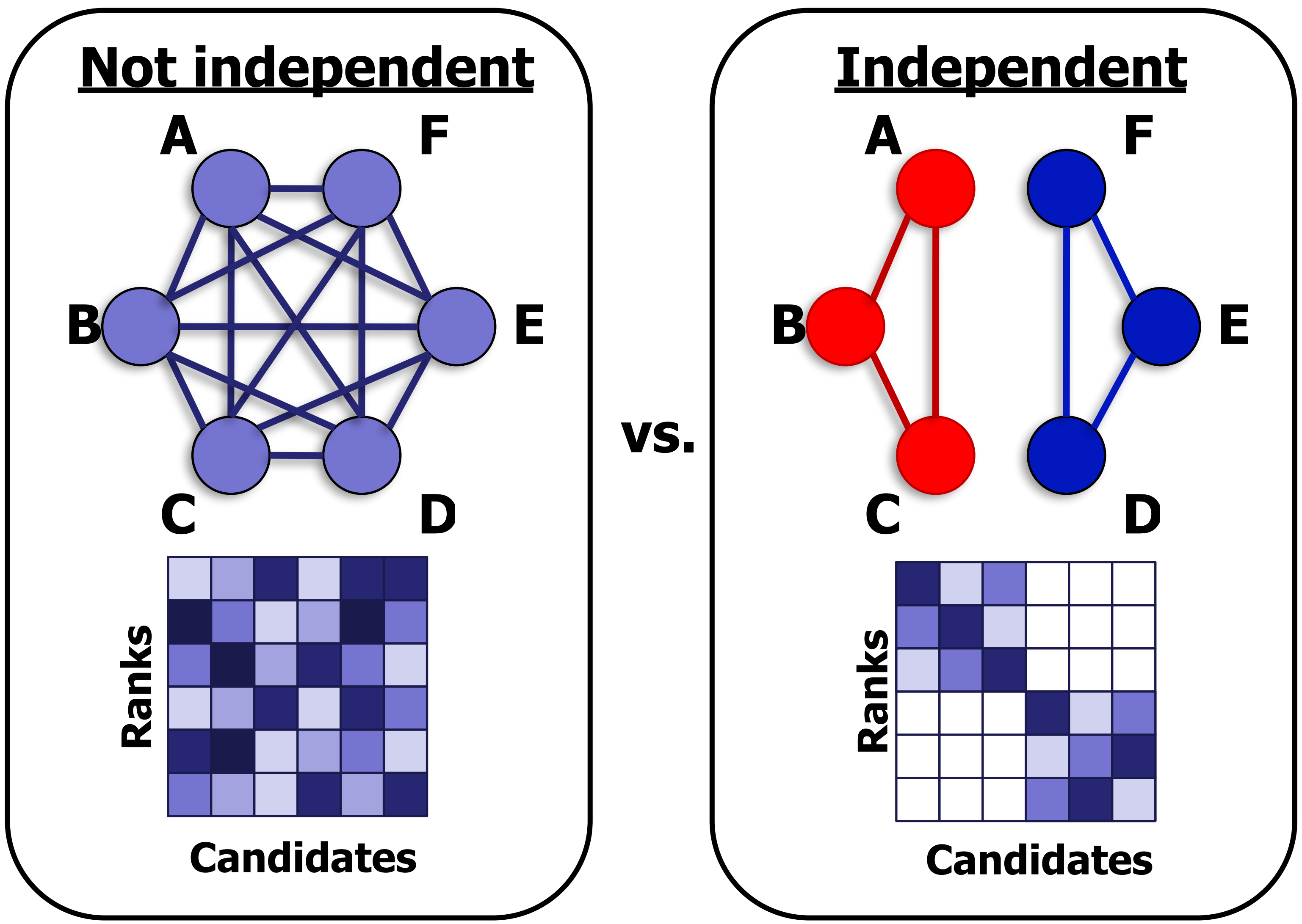
Probabilities in permutations

- We have n items, so $n!$ permutations
  - p(123) = 0.1
  - p(132) = 0.2
  - p(231) = 0.1
  - p(213) = 0.3
  - p(312) = 0.1
  - p(321) = 0.1
  - $\sum_{\sigma \in S_n} p(\sigma) = 1$

Permutations

- UMDA is based on independence. How does this translate to permutations?
- Matchings, routes, orders, rankings, …
    - Let’s think about rankings. They code preferences

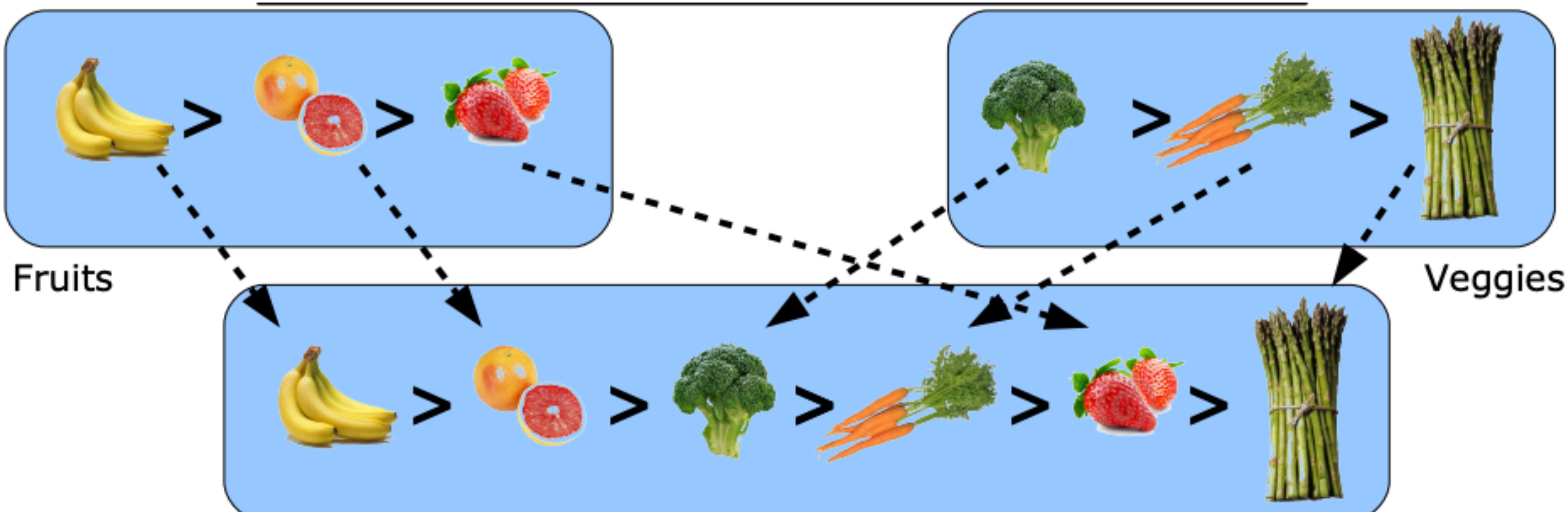
Independence



- 2 3 1 4 5 6
- 1 3 2 4 5 6
- 3 2 1 6 5 4
- 3 1 2 5 6 4

Riffled independence

- 3!+3! +interleaving< 6!



Conclusions

- Crossover does not adapt well to permutations in certain conditions
- Independence does not adapt well to permutations
- An alternative is to use EDA with specific distributions
  - Need to define distributions for permutations
  - Need to operate with them

Probability models for permutation problems

Symmetric group

- Composition
  - Examples
- Inverses
- Identity permutation is 0,1,2,3,4,…

Distributions for permutation

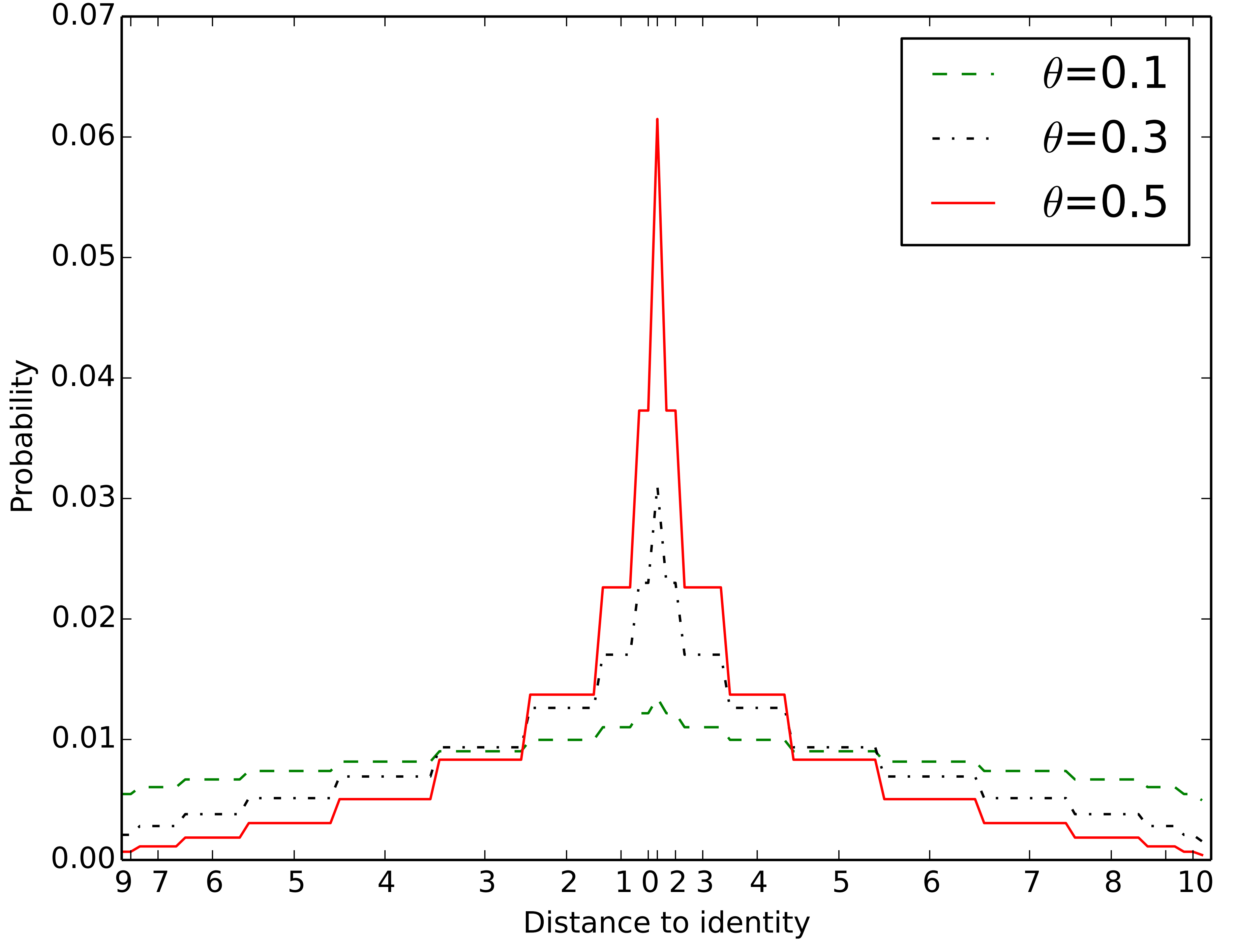
- Mallows model is an exponential location based model for permutations
- $p(\sigma) \propto exp(-\theta d(\sigma_0, \sigma))$
  - σ_0 and θ parameters instead of $n!$
  - How does it relate to the Normal distribution?
- Why do we need models?

Mallows model

- MM gives larger probability to permutations that are close to the central permutation σ_0

- The dispersion parameter θ controls the sharpness of the distribution. When 0 we have the uniform distribution

MM as exponential



Distances and rankings

- $\sigma(i)$ is the rank of item i
- Kendall's-τ counts the number of pairs of items that are ranked in different order in both rankings/permutations
  - Formalization and examples
- Hint: make one of the permutations to be the identity

Sampling

- Sampling is not an easy problem.
- How do you generate a random permutation? Not MM, what happens when you run np.random.permutation(n)?
  - Try generating a permutation at distance 3 u.a.r. Hint: Kendall's-τ from e counts the number of adjacent swaps
- Check with the code
  - Generate several samples with $\theta \in \{0.001, 1, 2, 3\}$

Learning

- We are given a sample of permutations and we want to estimate the parameters
- A common approach for learning in maximum likelihood estimation
- In MLE for the Mallows model the learning process is divided into
  - Learn the central ranking
  - Learn the dispersion

Exercise

- How many samples do we need to learn a distribution?
- Try different values for permutation length and variance

FSP

- Scheduling m jobs on n machines. A job consists on m operations and the jth operation of each jobmust be processes in machine j for a specified time. A job can start on the jth machine when its j-1 th operation has finished on the j-1 machine and the machine is free

- Minimize the makespan

More distances

- Hamming, counts the number of pairwise disagreements
- Cayley counts the number of swaps to convert a permutation into the other
- Ulam counts the length of longest common subsequence
- http://www.sc.ehu.es/ccwbayes/members/ekhine/papers/PerMallows.pdf