GLOBAL AND MULTI-OBJECTIVE OPTIMIZATION

Luca Manzoni

ESTIMATION OF DISTRIBUTION ALGORITHMS

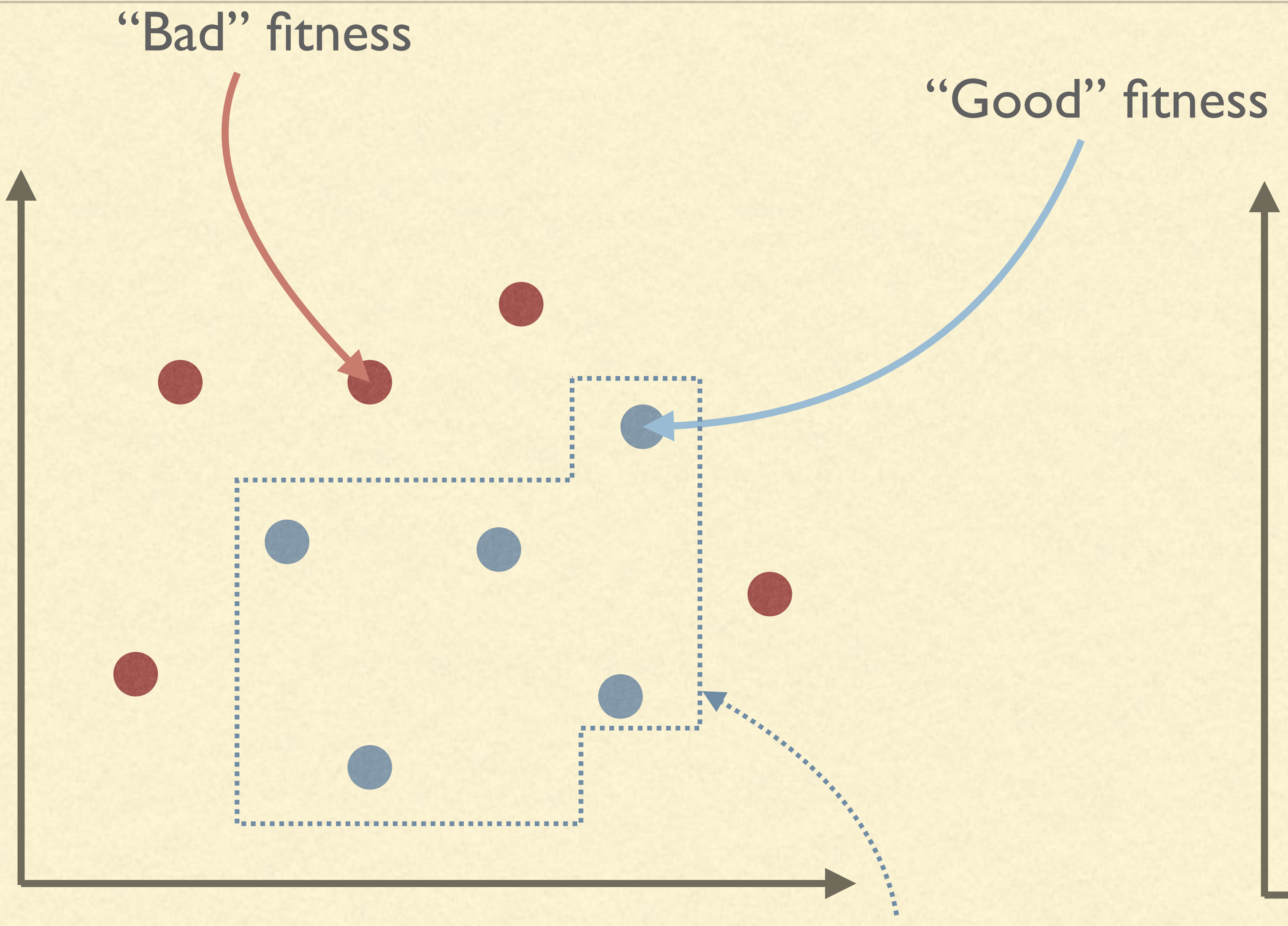
IMPLICIT AND EXPLICIT MODELS

- Selection, recombination (crossover), and mutation are used to sample the search space
- We may decide to build an explicit model of the space and use it to sample new solutions
- This is an indirect way of performing selection, recombination, and crossover by using an explicit model
- The new solutions will then be used to update our model (repeat as many time as desired)

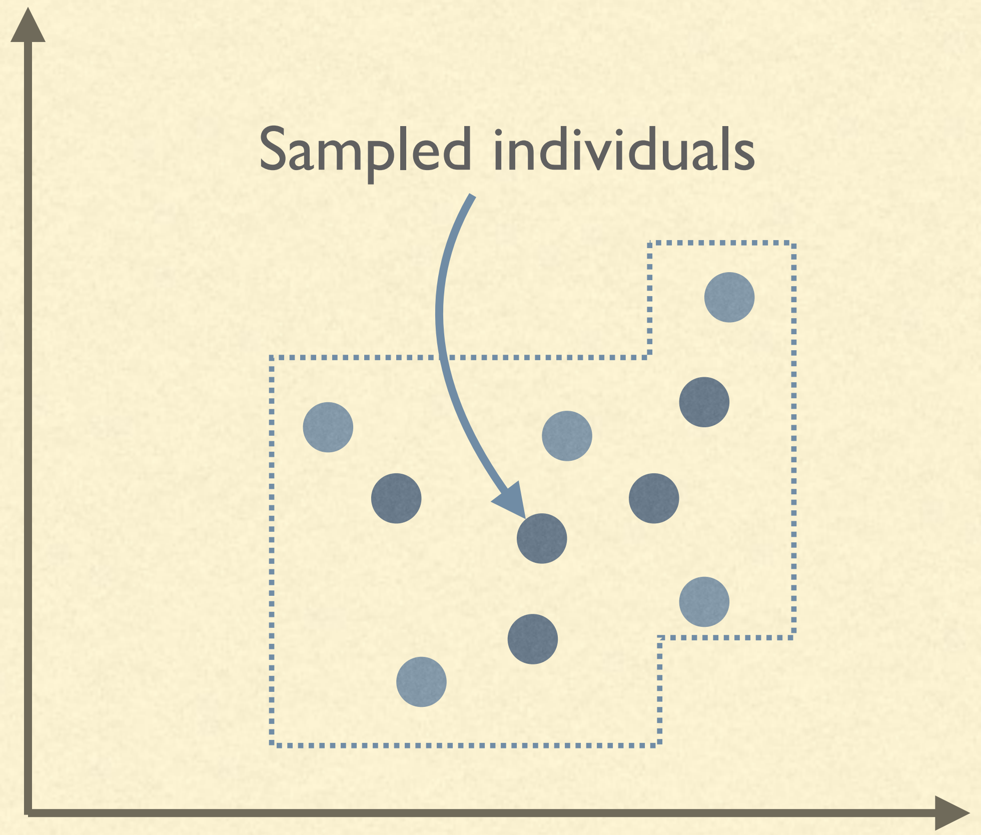
MODEL FITTING BY CLASSIFICATION

- We want to sample new solutions in the “good” areas of the search space
- The idea is to use a **classifier** (e.g., decision trees, SVM, etc) to classify a region of the search space as “good” or “bad” based on the current population
- We then use the information provided by the classifier to generate a new population
- The most famous approach are the **Learnable Evolution Models** (LEM)

CLASSIFICATION



Area of the search space classified as "good" by a decision tree



"Bad" individuals are removed and replaced by new individuals sampled from the "good" parts of the search space

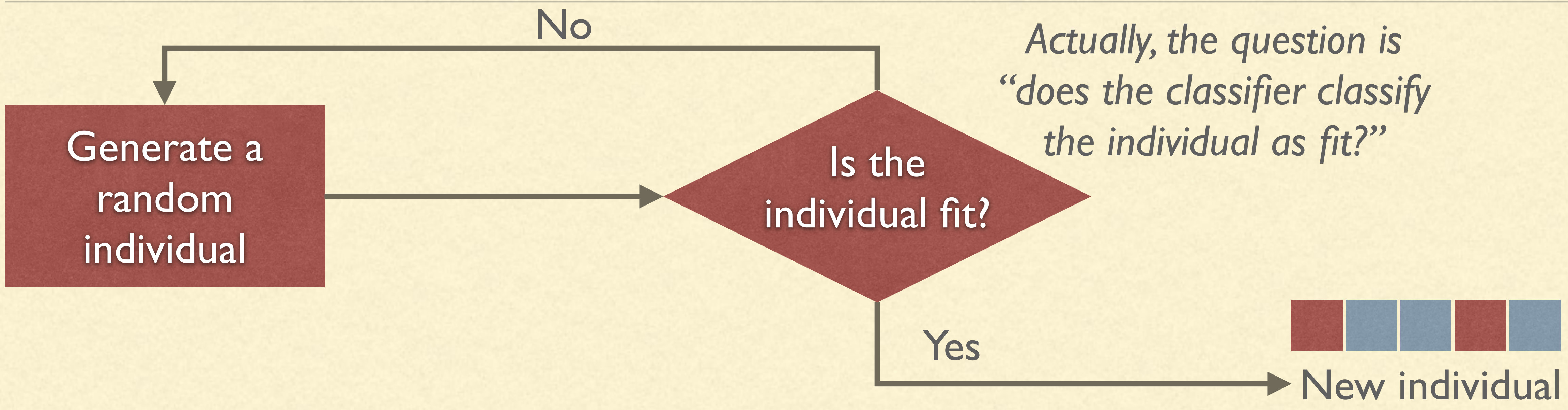
LEARNABLE EVOLUTION MODEL

- Perform some evolutionary steps
- Divide the population in “fit” and “unfit” individuals
- Train a classifier (e.g., SVM, Neural Networks, Decision Trees) to distinguish between fit and unfit individuals
- Remove the unfit individuals and replace them with individuals classified as “fit”
- Go to the first step unless some termination criteria has been met

GENERATIVE VS DISCRIMINATIVE MODELS

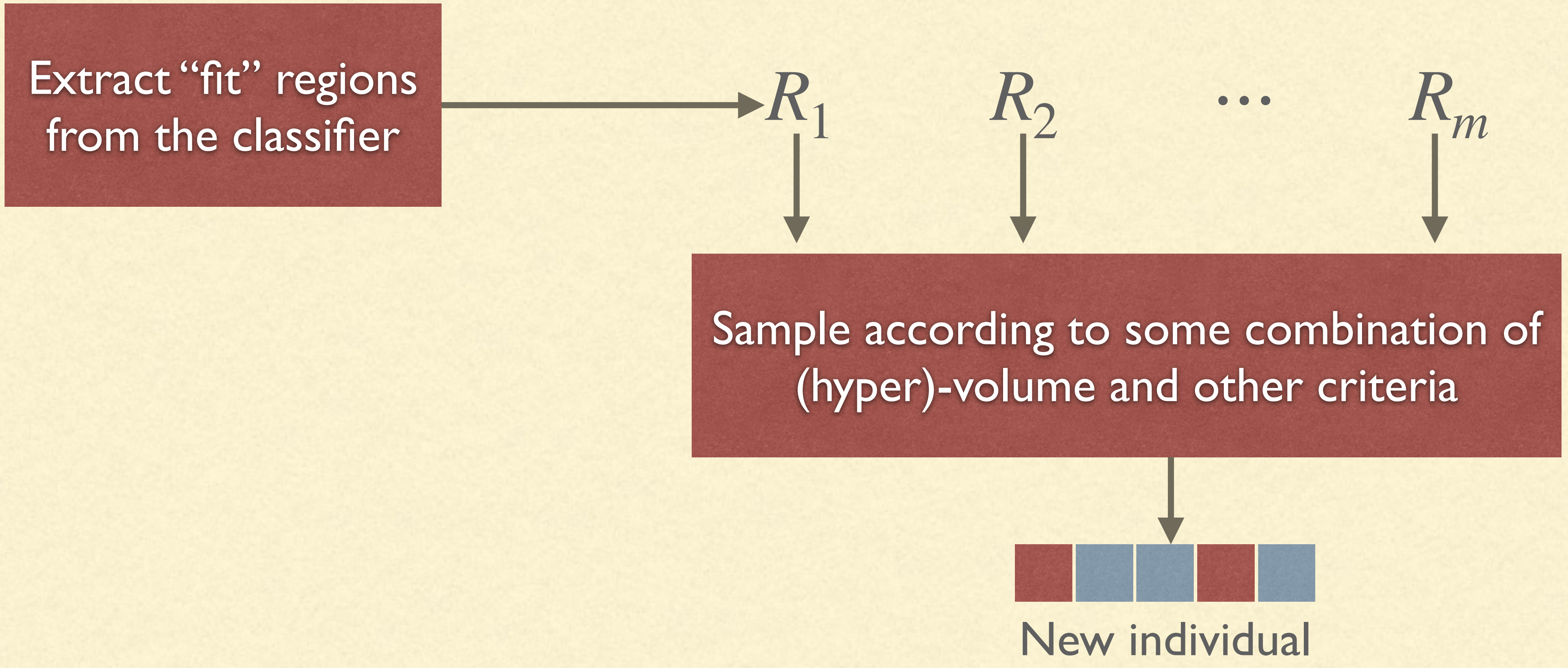
- We can divide the models in two classes:
  - **Generative**. They can be used directly to generate new individuals
  - **Discriminative**. Given an individual they can discriminate if the individual is "good" or "bad"
- Classifier are generally discriminative models, hence we usually employ them with *rejection sampling*.

REJECTION SAMPLING


No
Generate a random individual
Is the individual fit?
Actually, the question is "does the classifier classify the individual as fit?"
Yes
New individual

- We can impose a limit on the number of tries
- After a certain amount of time we have the problem of keeping the number of tries under a reasonable amount
- Most of the search space might be classified as "unfit", so this sampling becomes too expensive
- We can easily extend it to deal with classifications that are not binary (weighted rejection sampling)

REGION-BASED SAMPLING


Extract "fit" regions from the classifier
R_1
R_2
$\cdots$
R_m
Sample according to some combination of (hyper)-volume and other criteria
New individual

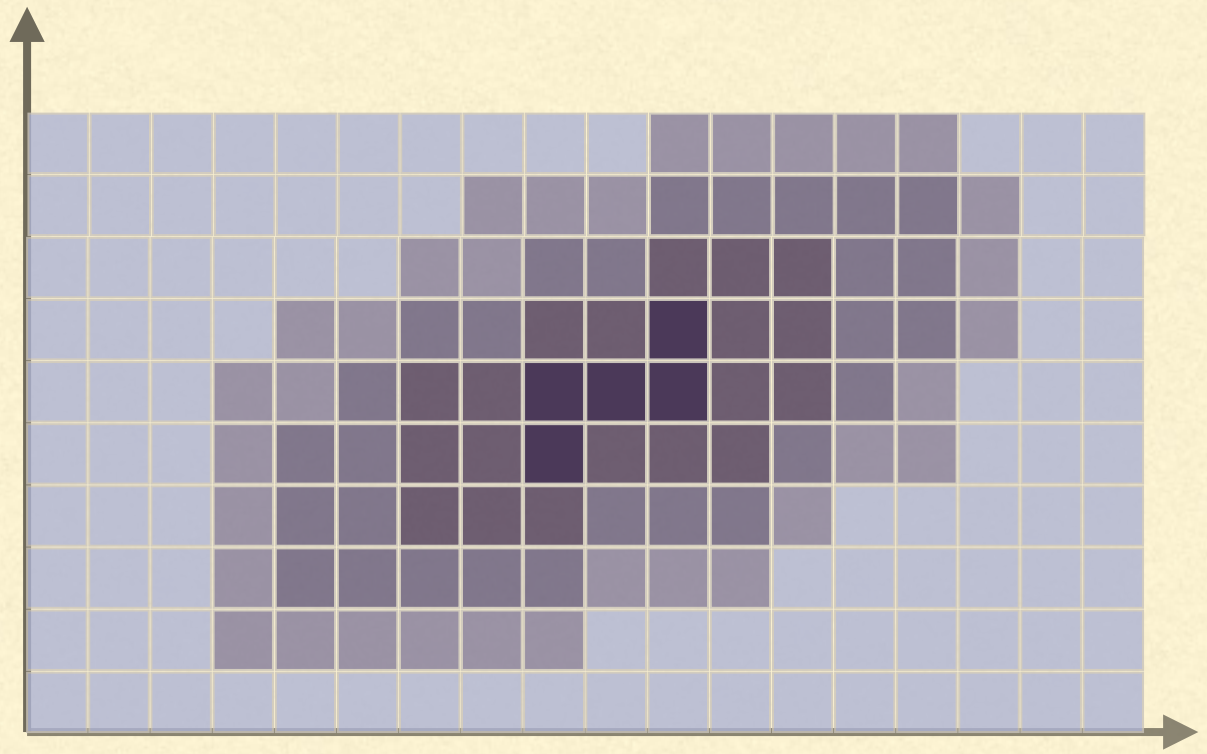
- Limited only by our ability to extract fit regions from the classifier
- E.g., perfectly possible with decision trees

WHAT ARE EDA?

- Sometimes called "probabilistic model-building genetic algorithm" (PMBGA)
- Instead of using an implicit model or a classifier, EDA use directly a probability distribution to sample the new solutions

HISTOGRAMS

The space can be partitioned into (hyper)cubes and we can compute the average fitness of the samples in each of them



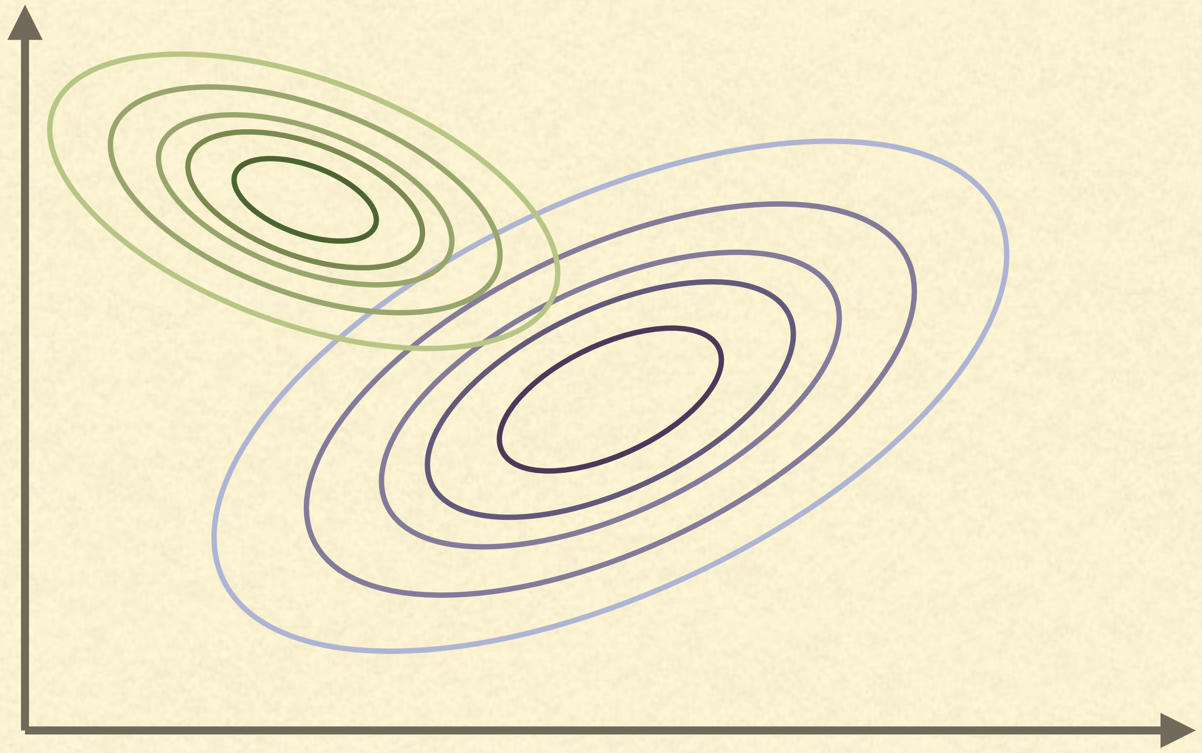
If we split each of the n dimensions in d intervals we have d^n hypercubes

Even for $d = 2$ and a small number of dimensions this is unfeasible

A more compact way of representing the fitness distribution should be used

MULTIPLE GAUSSIAN DISTRIBUTIONS

Instead of representing the distribution as an histogram, we can represent it as the sum of a (fixed) number b of Gaussian distributions in n dimensions

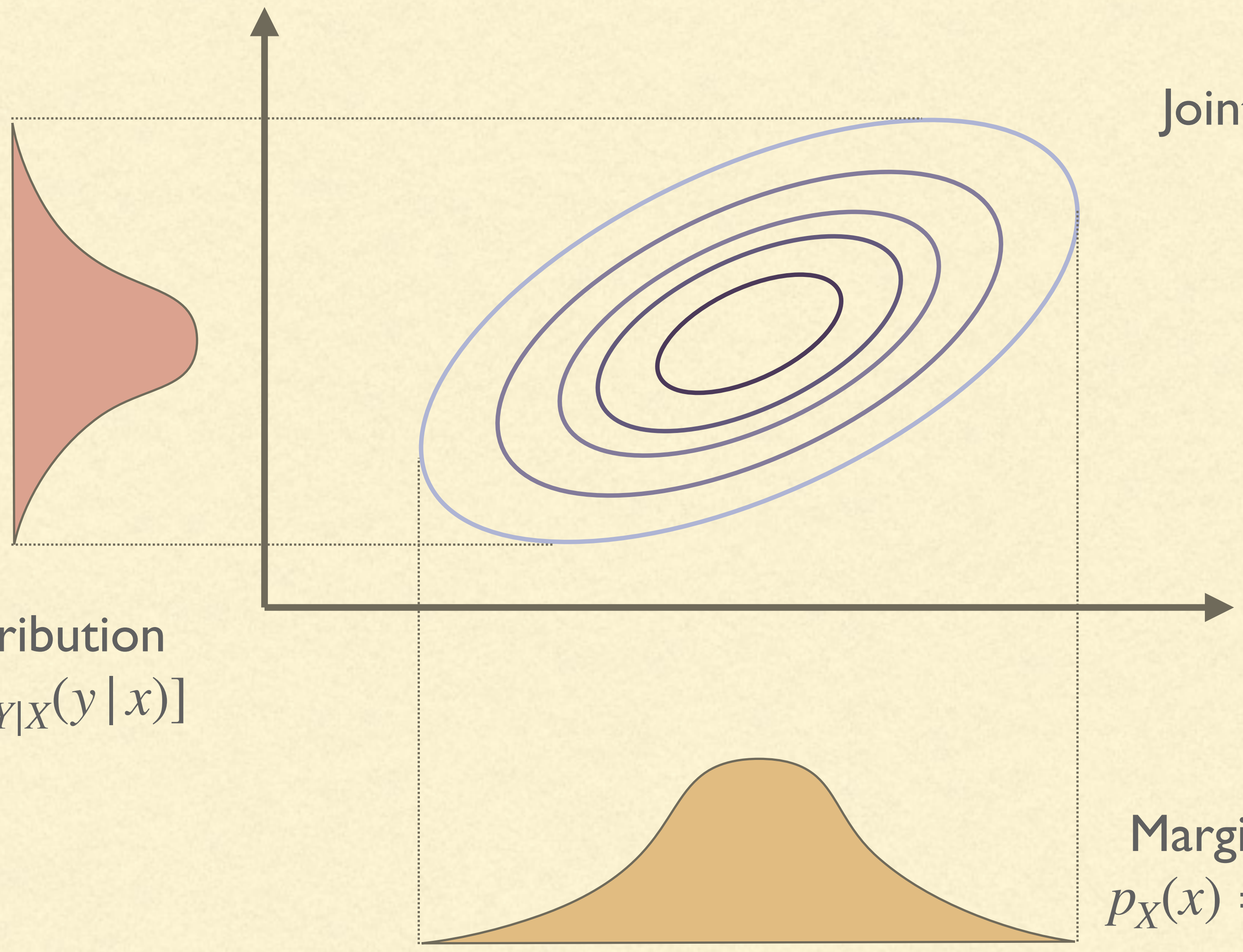


More in general, we can use a collection of "known" distributions and sum them

Each Gaussian distribution in n dimensions is determined by:
- The mean vector $\overrightarrow{\mu}$ of length n
- The covariance matrix Σ of size n^2

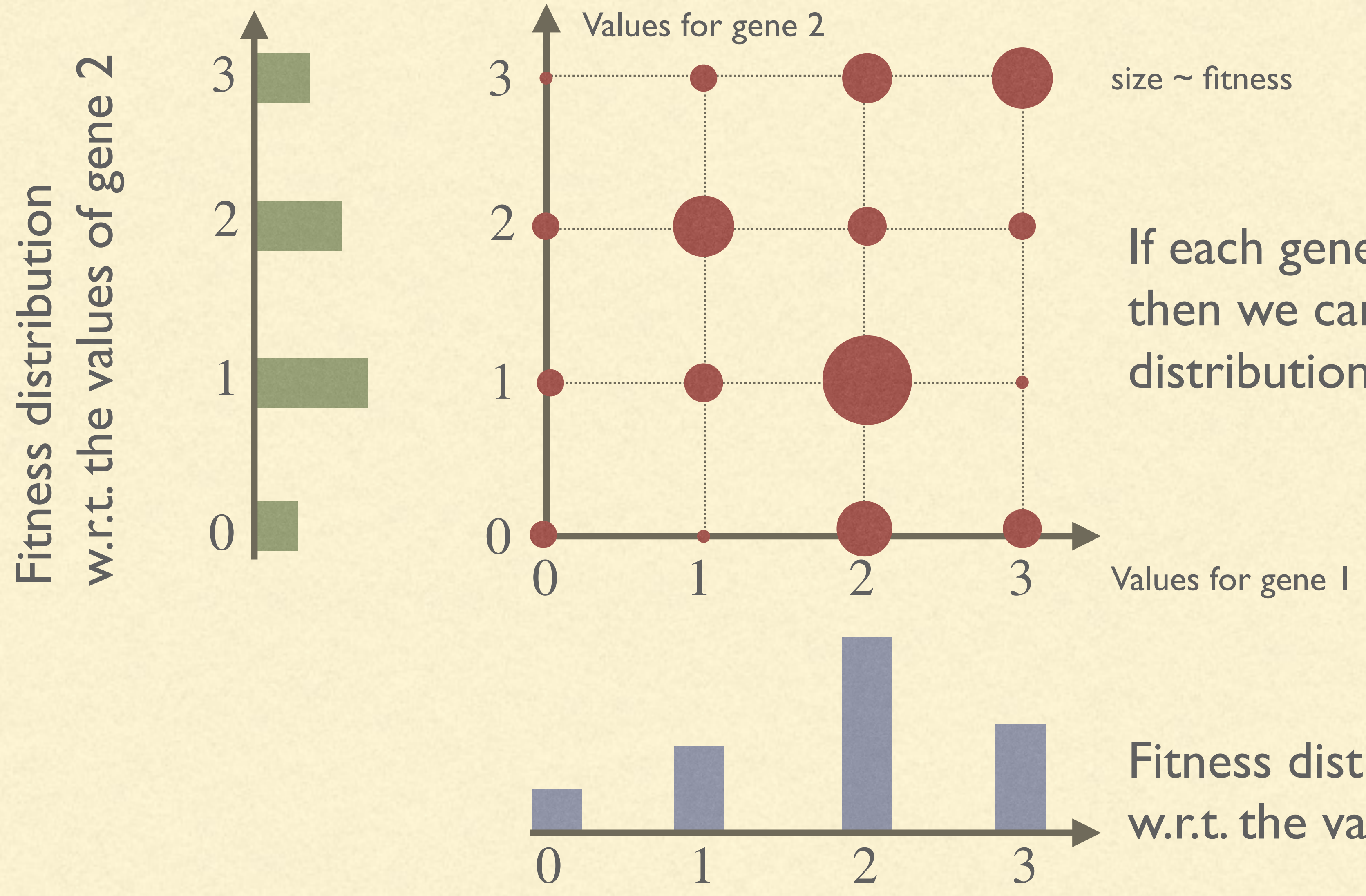
Hence, to represent the distribution as the sum of b Gaussians, we need $b(n + n^2)$ space

MARGINAL DISTRIBUTIONS



Each (one-dimensional) marginal distribution of b Gaussians requires only a mean and a variance, hence the total space used is $2bn$

FINITE DISCRETE SPACES



If each gene can have at most k values, then we can represent each marginal distribution as a vector of length $k - 1$

Fitness distribution w.r.t. the values of gene 1

For Boolean spaces we can represent the marginal distributions as a n-dimensional vector with values in $[0,1]$

UNIVARIATE EDA

- One standard approach with EDA is to marginalise everything to have only distributions in one dimension
- We are actually assuming that the value of each gene can be determined independently from the values of the others genes
- We will see two univariate EDA algorithms:
  - Population-based incremental learning (PBIL)
  - Compact Genetic Algorithm (cGA)

POPULATION-BASED INCREMENTAL LEARNING

- For dimension n the algorithm starts with n marginal distributions (initially uniform) $D_1, \ldots, D_n$
- New individuals are sampled picking one gene from each distribution
- Update the marginal distribution according to the fitness of the sampled individuals
- Repeat as needed

PBIL: UPDATE

- Keep the b fittest individuals in the newly sampled population (i.e., truncated selection)
- For each gene j, let N_j be the distribution of values of gene j in the b fittest individuals
- Update $D_j \leftarrow (1-\alpha)D_j + \alpha N_j$ with $\alpha \in [0,1]$
- α allows to change the distribution *gradually*

PBIL: EXTENSIONS

- PBIL can be extended to work in continuous spaces (e.g., $\mathbb{R}^n$)
- One approach is to discrete each marginal distribution into k "buckets"
- Instead of using a discrete distribution we can use a Gaussian for each marginal distribution
- In that case each distribution is defined by μ_{D_j} and $\sigma^2_{D_j}$ and the updating rules must modify them

PBIL: EXTENSIONS

- Let P be the current population *after the truncated selection* and $P_{i,j}$ the value of gene j in individual i

- We can compute the following values on P
$$\mu_{N_j} = \frac{1}{|P|} \sum_{P_i \in P} P_{i,j} \qquad \sigma^2_{N_j} = \frac{1}{|P| - 1} \sum_{P_i \in P} (P_{i,j} - \mu_{N_j})^2$$

- And the update of the mean and variance of each marginal distribution is performed as
$$\mu_{D_j} \leftarrow (1-\alpha)\mu_{D_j} + \alpha\mu_{N_j} \qquad \sigma^2_{D_j} \leftarrow (1-\beta)\sigma^2_{D_j} + \beta\sigma^2_{N_j}$$

COMPACT GENETIC ALGORITHM

- cGA operates *only* over Boolean spaces
- cGA update the marginal distributions in steps of a fixed size
- Instead of computing a new probability distribution, the individuals in the population are compared in pairs
- For each pair P_i, P_k of individuals, if P_i is fitter than P_k and they differ at gene j, then we shift the distribution D_j to produce the value of gene j in P_i more often

COMPACT GENETIC ALGORITHM

- Let $\frac{1}{d}$ be out **discretisation** value (think of it as a "learning rate"). This will be the "step" used to change out distributions
- We identify D_j with the probability of generating a *one* for gene j
- Generate a population P of individuals in which each gene is sampled according to D_j

Repeat as needed

- For each pair P_i, P_k of individuals, if P_i is fitter than P_k and they differ at gene j, then we shift the distribution D_j to produce the value of gene j in P_i more often (more details later)

COMPACT GENETIC ALGORITHM

- For each pair of individuals $P_i, P_k \in P$
  - Let U be the fittest between P_i and P_k and V the other one
  - If $U_j \neq V_j, U_j = 0,$ and $D_j > 0$
    - $D_j \leftarrow D_j - \frac{1}{d}$ *shift the distribution toward zero*
  - If $U_j \neq V_j, U_j = 1,$ and $D_j < 1$
    - $D_j \leftarrow D_j + \frac{1}{d}$ *shift the distribution toward one*

ISSUES OF UNIVARIATE EDA

- Univariate EDA assume that the distribution of each gene can be found independently from all the other genes (i.e., no linkage between genes)
- This is usually false (otherwise we would be able to optimise each gene independently)
- Univariate EDA can get stuck in local optima due to this
- **Multivariate EDA** allow to model interaction between genes

BAYESIAN OPTIMISATION ALGORITHM

- Among the different multivariate EDA, one is the Bayesian Optimisation Algorithm (BOA)
- Similar to PBIL where, instead of using marginal distributions, a Bayesian network is used to generate the samples and updated at every generation
- Pelikan, Martin, David E. Goldberg, and Erick Cantú-Paz. *"BOA:The Bayesian optimization algorithm."* Proceedings of the genetic and evolutionary computation conference GECCO-99. Vol. 1. 1999.