Neural networks for population genetics: demographic inference and data generation

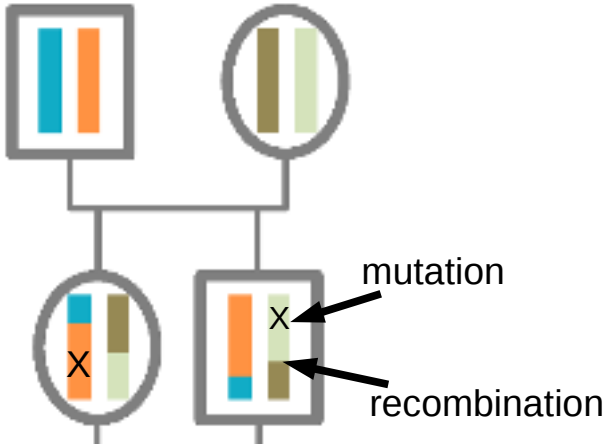
Flora Jay
Théophile Sanchez, Jean Cury, Guillaume Charpiat
Burak Yelmen, Aurelien Decelle, Linda Ongaro, Davide Marnetto, Francesco Montinaro, Corentin Tallec, Cyril Furtlehner, Luca Pagani

Institute of Genomics, Tartu, Estonia
Laboratoire de Recherche en informatique

Overview

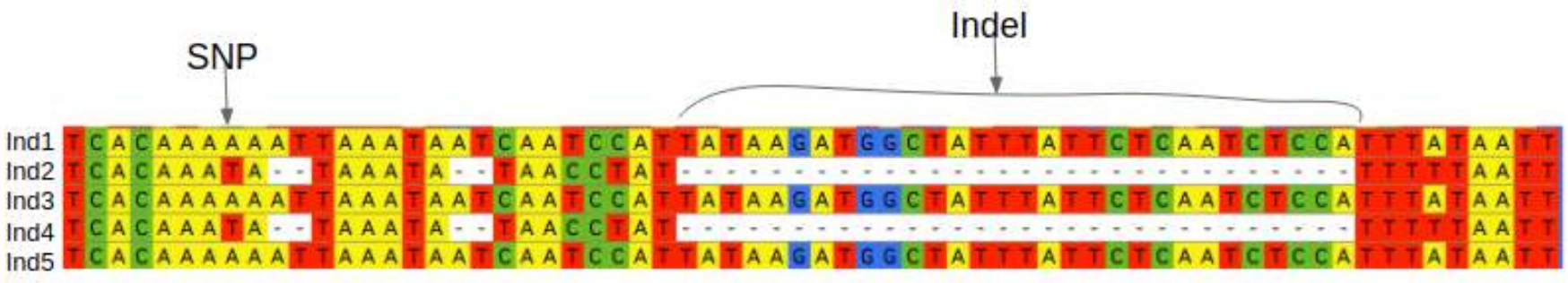
- Population Genetic Context
- Artificial Genomes project – based on generative networks
- Demographic inference project

Population Genetics



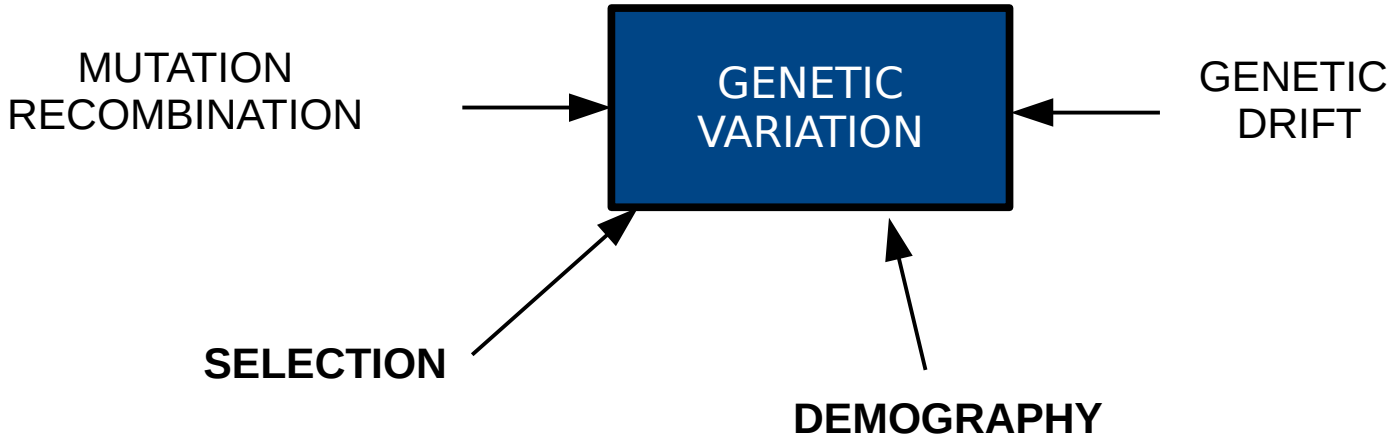
Population Genetics - Data

Recombination + mutation create diversity



© A Branca

Population Genetics



Population Genetics

MUTATION
RECOMBINATION

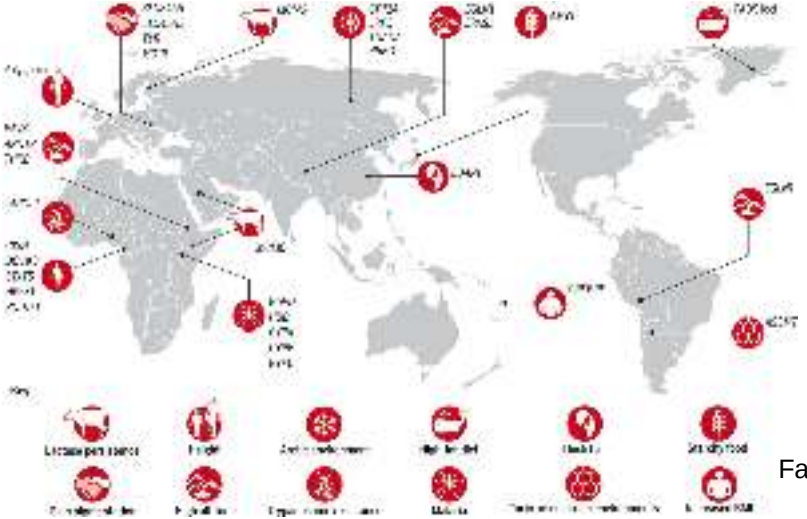
GENETIC VARIATION

GENETIC DRIFT

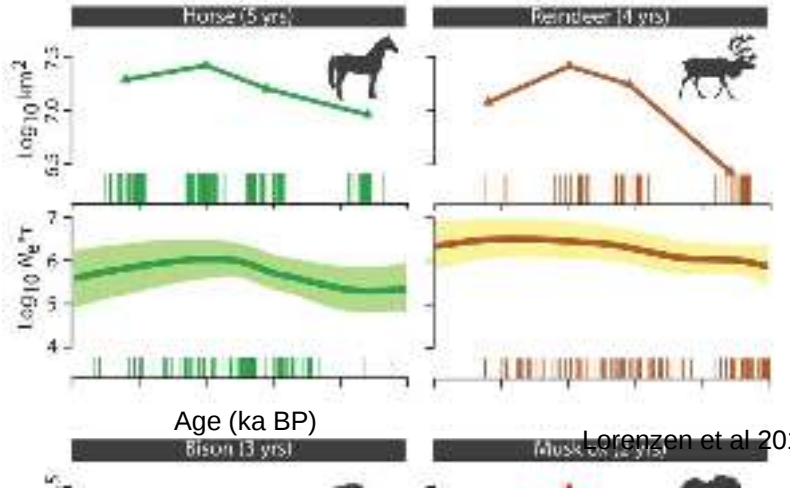
SELECTION

INFERENCE

DEMOGRAPHY



Fan et al 2016



Lorenzen et al 2011

Projects

(1) Unsupervised learning for learning the high dimensional distribution of existing genomic datasets

- Generative Adversarial Networks and Restricted Boltzmann Machines
- **Creating artificial genomes** that bear the characteristics of real ones

(2) Supervised learning for **demographic inference** from present-day genomic data

- Approximate Bayesian Computation and Deep Neural Networks
- Designing architectures tailored for population genomic data

Why Artificial Genomes (AGs)?

- Huge amount of genomic data but many datasets are **private** → loss of information (held by companies, government, institutions)
- In particular some populations are underrepresented in public data → decreased resolution in studies (Sirugo et al 2019)

- Could we **create Artificial Genomes to augment public datasets**?
- Do AGs retain important **characteristics**?
- Can we show that AGs are useful for **population genetics tasks** (detecting selection, gwas, imputation, ancestry inference,...)?

HOW? With generative models (unsupervised learning)

Data with no label

Goal Generate samples having the same distribution as the data

Training data ~ $p_{data}(x)$

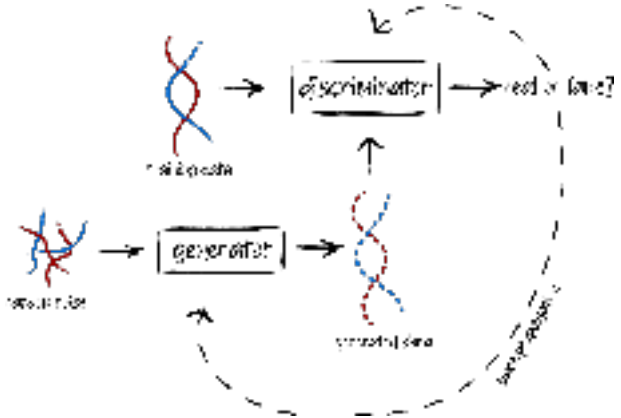
(distribution unknown)

Generated samples ~ $p_{model}(x)$

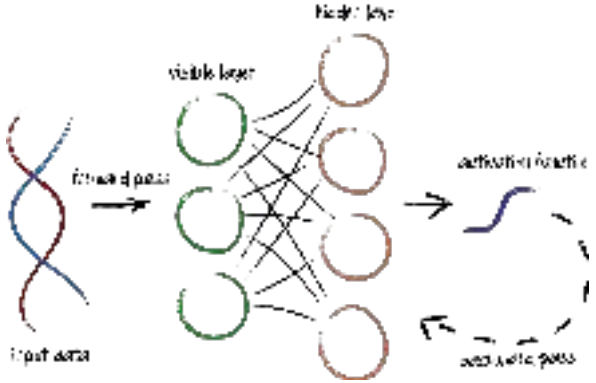
How?

With **generative models**

Generative Adversarial Network (GAN)



Restricted Boltzmann Machine (RBM)



Previous works

- None in population genetics
- Some in genetics often for proteins (Killoran et al 2017, Davidsen et al 2019, Liu et al 2019, Tubiana et al 2019, Shimagaki and Weigt 2019),

Generative Adversarial Networks (GANs)

Goodfellow et al 2014

Karras et al 2018 *proGAN*

Generative Adversarial Networks (GANs) Goodfellow et al 2014

- Generate samples from $p_{model}(x)$ without explicitly defining it
  i.e. sample from a complex high-dimensional unknown distribution

- Solution: sample from a simple distribution (random noise) and learn the transformation to the real data distribution $p_{data}(x)$

- Use a Neural network to learn this transformation

Generative Adversarial Networks (GANs) Goodfellow et al 2014

- Generate samples from $p_{model}(x)$ without explicitly defining it
  i.e. sample from a complex high-dimensional unknown distribution

- Solution: sample from a simple distribution (random noise) and learn the transformation to the real data distribution $p_{data}(x)$

- Use a Neural network to learn this transformation

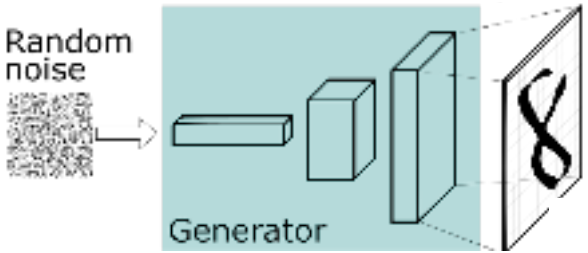


Image credit: Thalles Silva

Generative Adversarial Networks (GANs) Goodfellow et al 2014

Generate samples from $p_{model}(x)$ without explicitly defining it

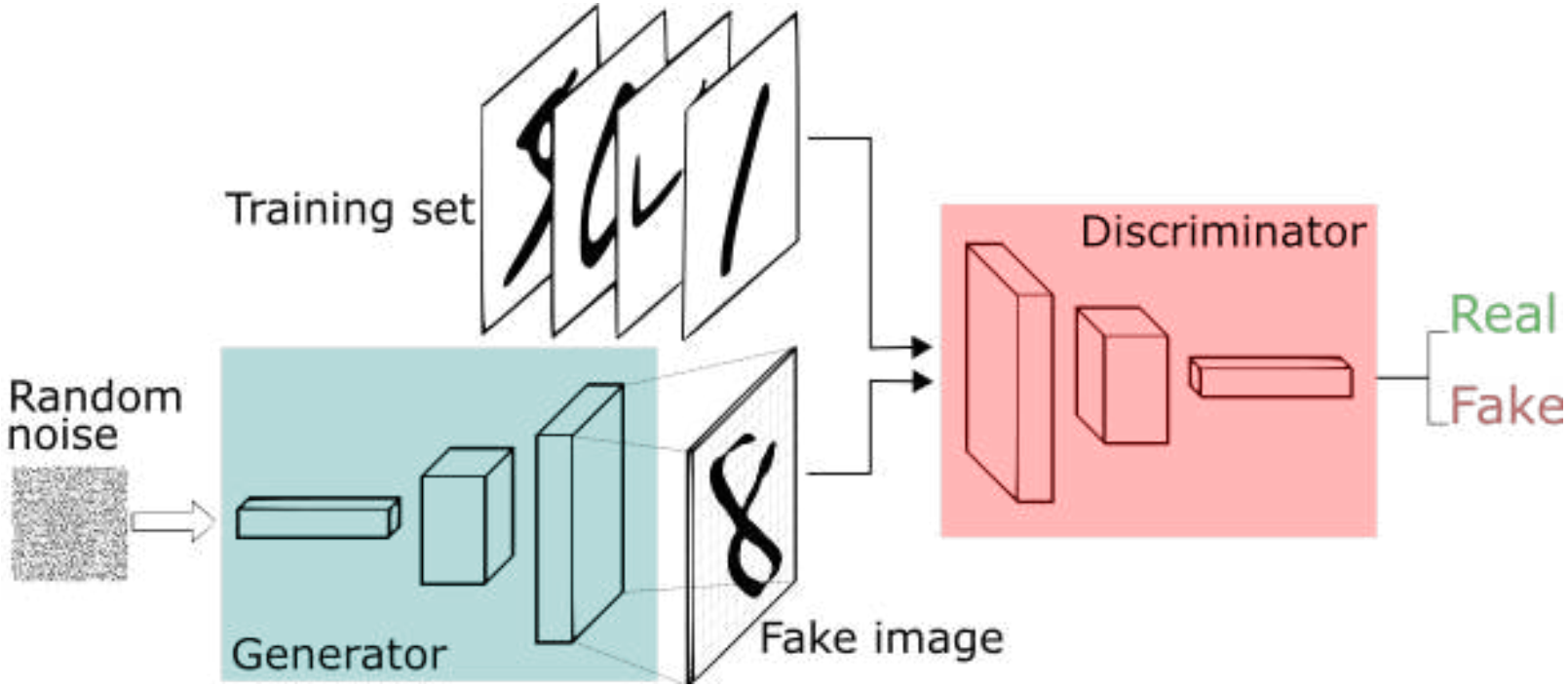


Image credit: Thalles Silva

GAN

Game between 2 players (2 neural networks)

- Generator creates images **as realistic possible to fool the Discriminator**
- Discriminator is trained to **beat the generator** (in a **supervised** fashion from Real and Fake images)

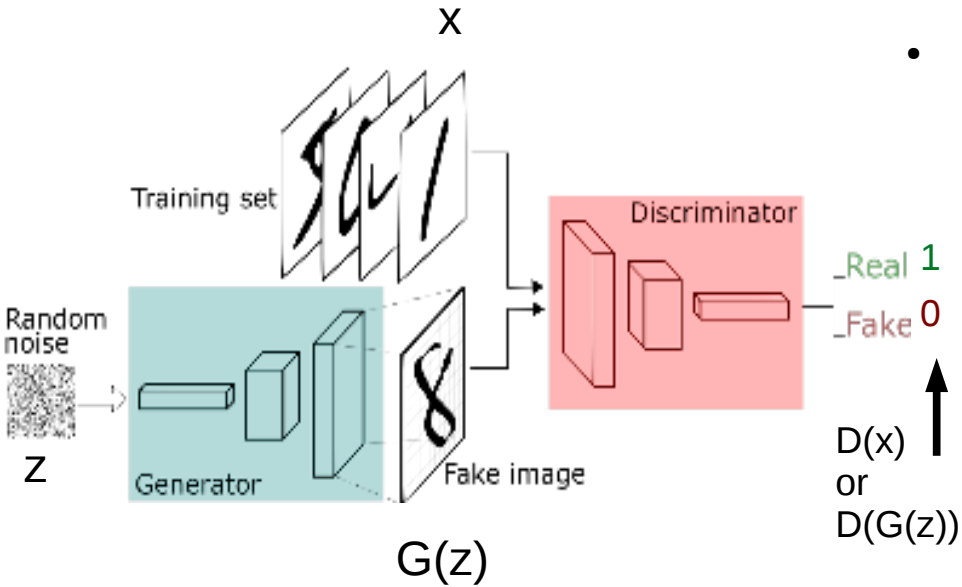
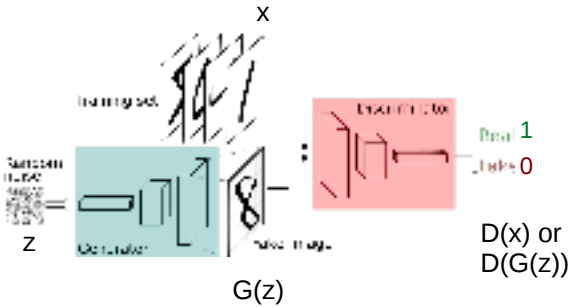


Image adapted from: Thalles Silva

GAN

Game between 2 players (2 neural networks)



- Discriminator wants to maximize its classification accuracy i.e. D(x) should be 1 and D(G(z)) should be 0

$$max_{\theta_d} E_{x \sim p_{data}}[\log D_{\theta_d}(x)] + E_{z \sim p(z)}[\log(1 - D_{\theta_d}(G(z)))]$$

Discriminator output for real data

Discriminator output for generated data

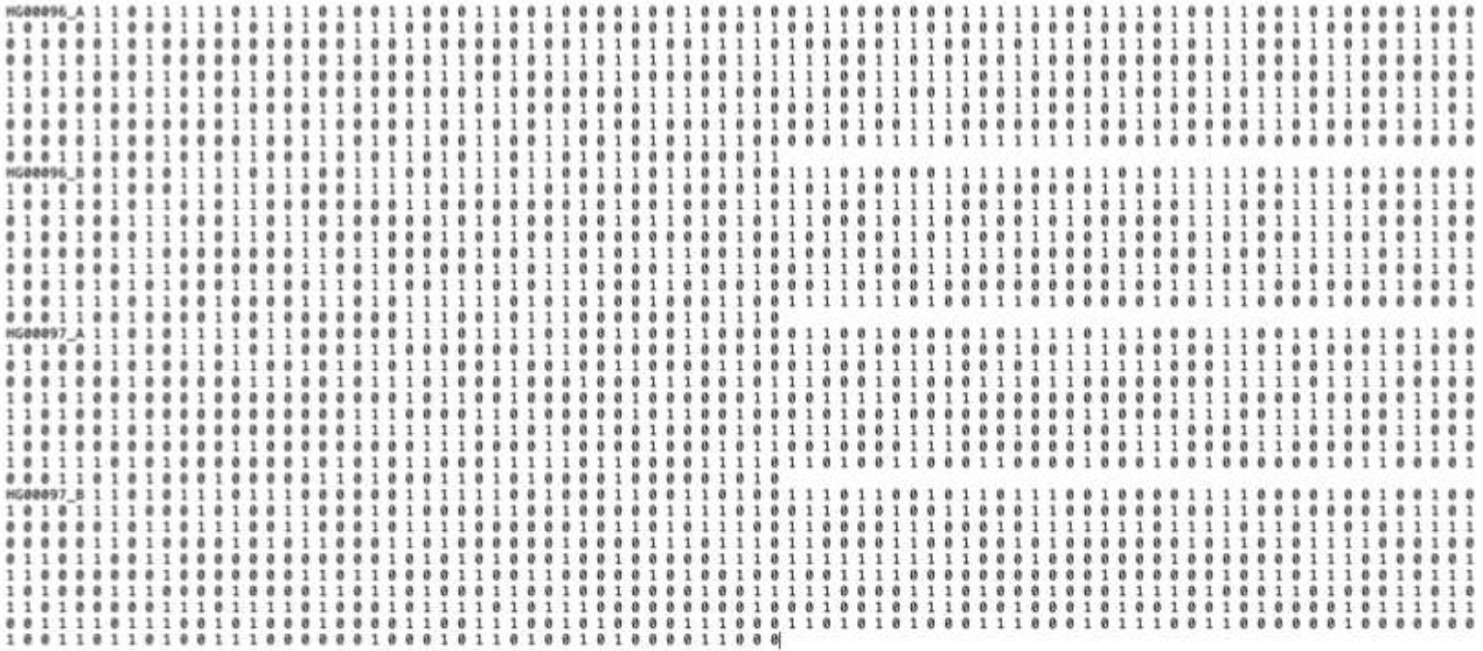
- Generator wants to maximize the likelihood of discriminator to be wrong i.e. D(G(z)) should be 1

$$max_{\theta_g} E_{z \sim p(z)}[\log(D(G_{\theta_g}(z)))]$$

Discriminator output for generated data

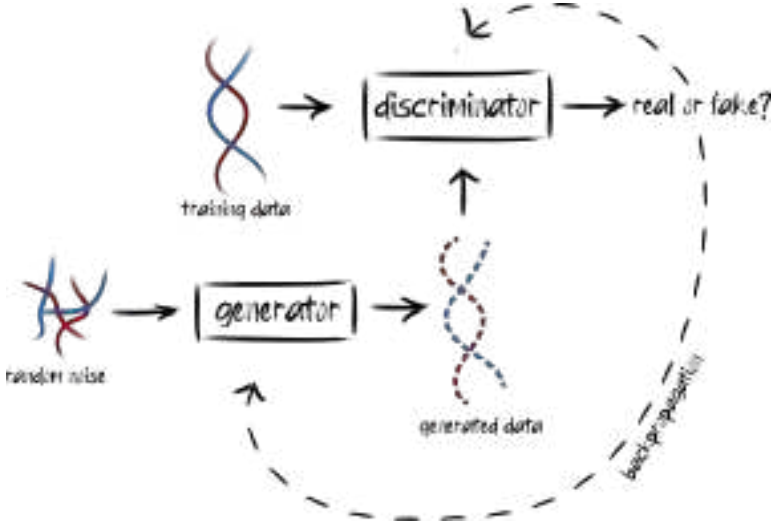
Our data, each row is an example for the GAN

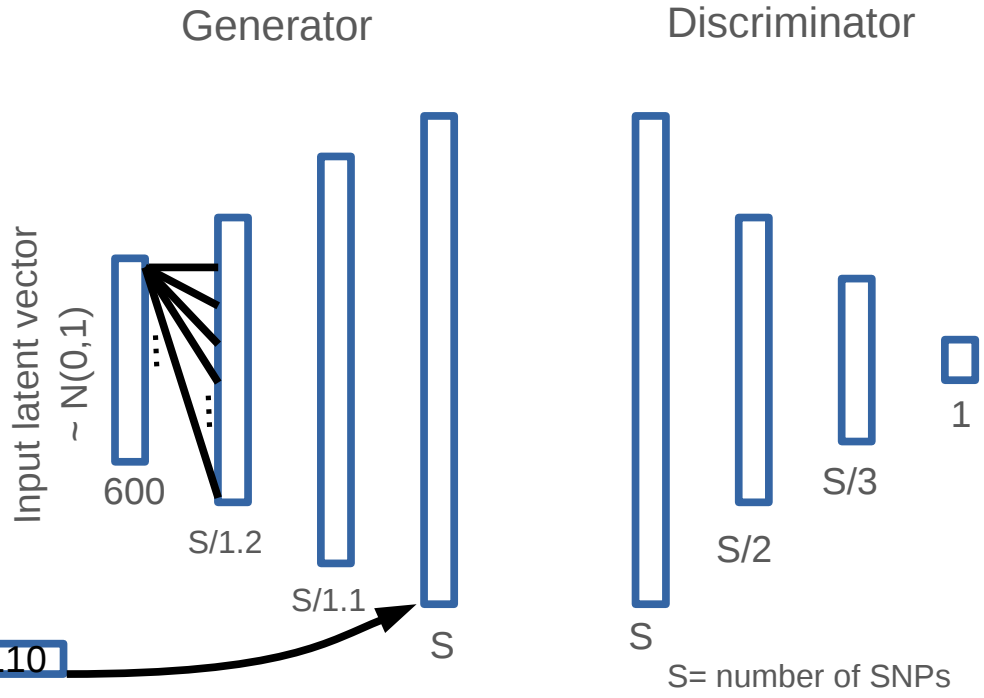
- Rows are individuals/haplotypes
- Columns are SNP positions -> features



GAN our implementation

Fully connected networks; with LeakyReLU activation (except for output layer)

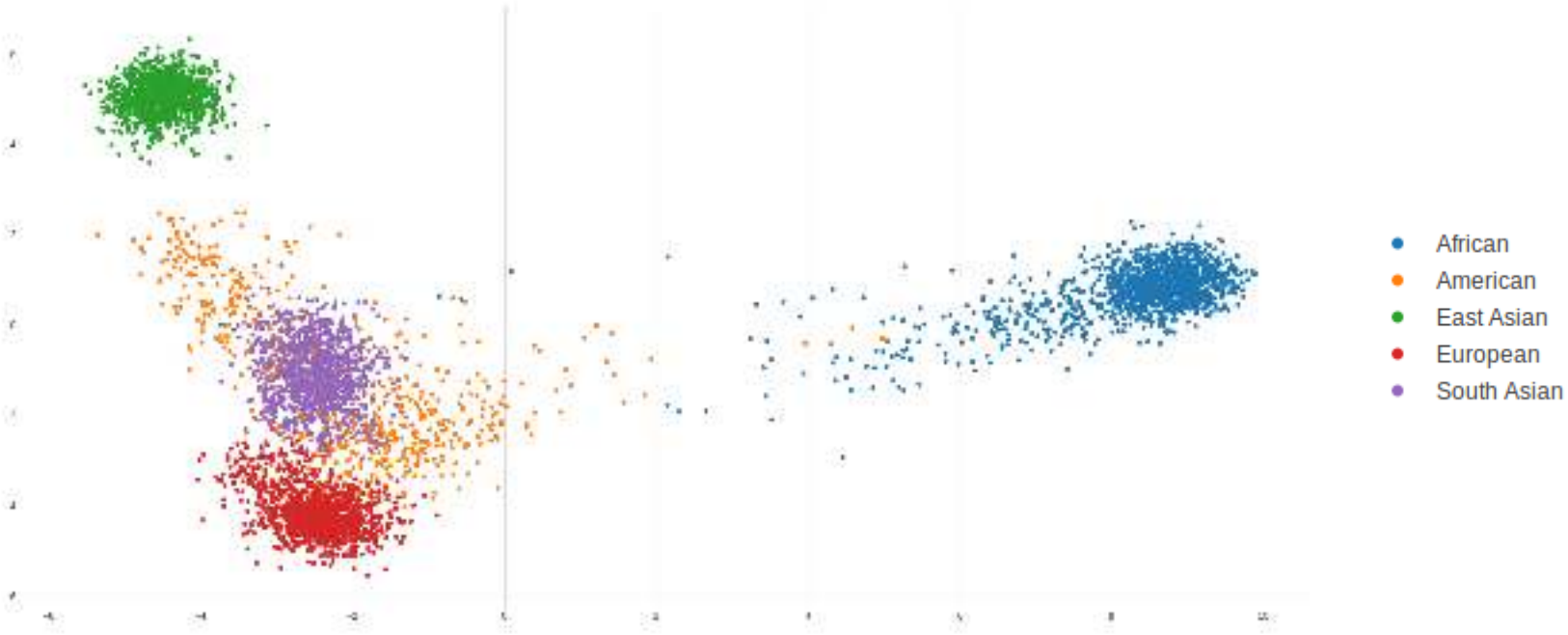




First example

1000Genomes human dataset 2504 individuals worldwide (1000 Genomes Project Consortium et al. 2015)

1000Genomes human dataset 2504 individuals worldwide (1000 Genomes Project Consortium et al. 2015)

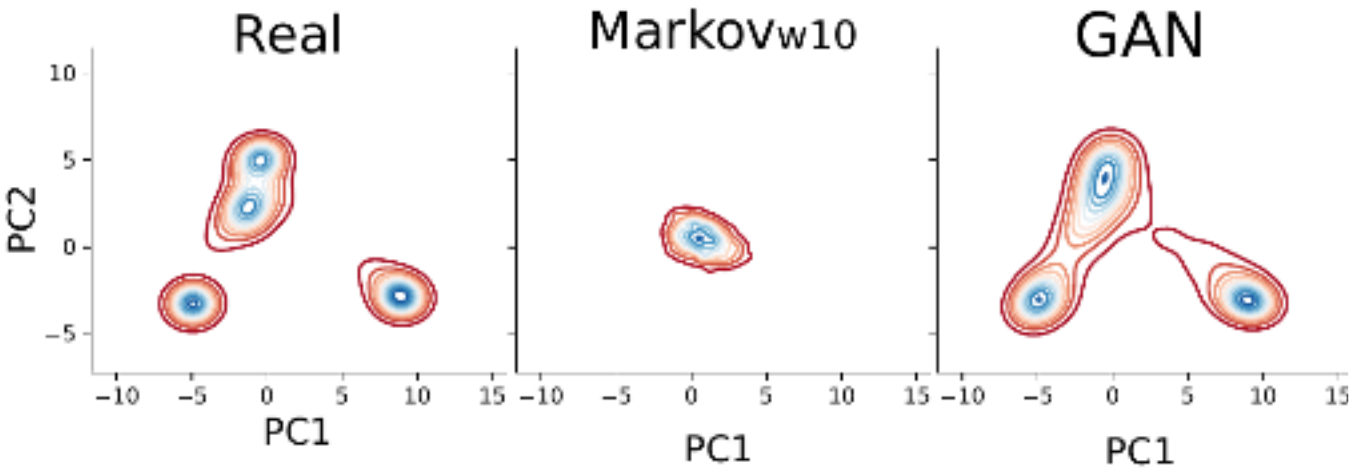


Spoiler (hidden population structure, ie relationships between indiv/lines)

PCA 2D density plots

1000 Genomes Panel (2504 individuals)

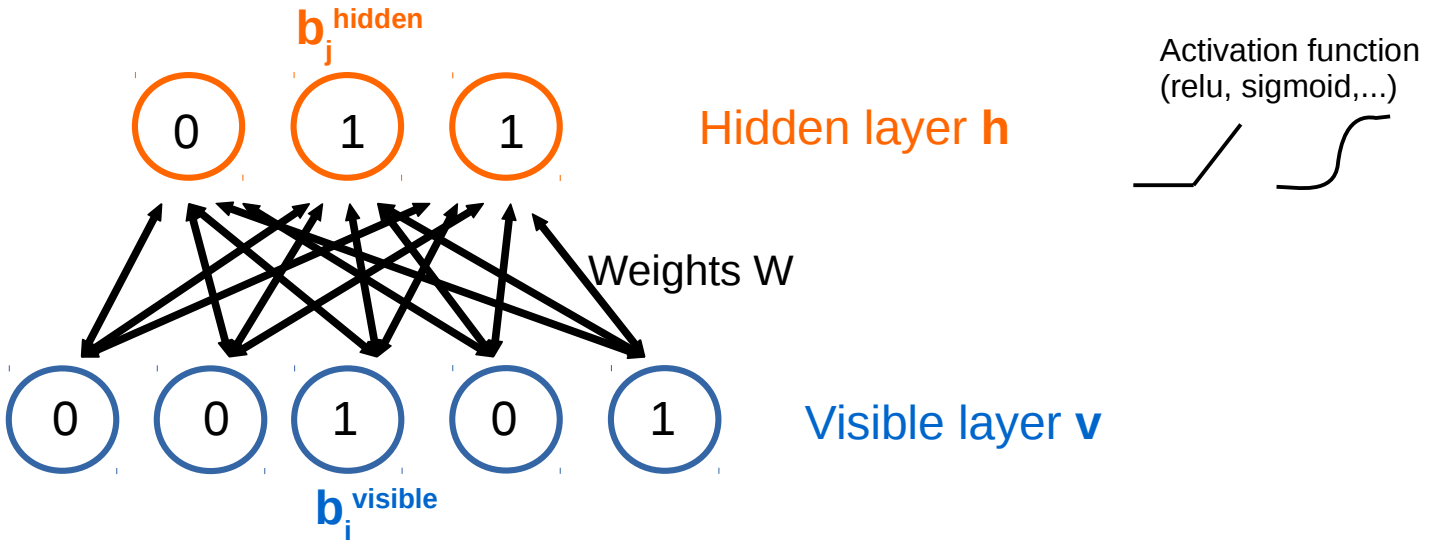
(a) 805 highly differentiated SNPs accross the genome (Colonna et al 2014)



Restricted Boltzmann Machines (RBMs)

Learn $p_{model}(x)$ that approximates $p_{data}(x)$ and generate samples from it

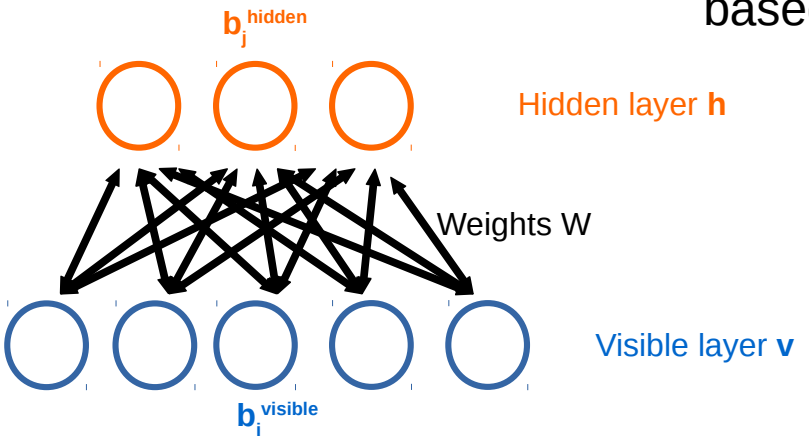
Two-layer network



Smolensky 1986; Teh and Hinton 2001; Hinton and Salakhutdinov 2006; Hinton 2007; Larochelle and Bengio 2008

Restricted Boltzmann Machines (RBMs)

Two-layer network



Probabilistic model of the joint distribution of **v** and **h** based on an **energy** function

Z partition function

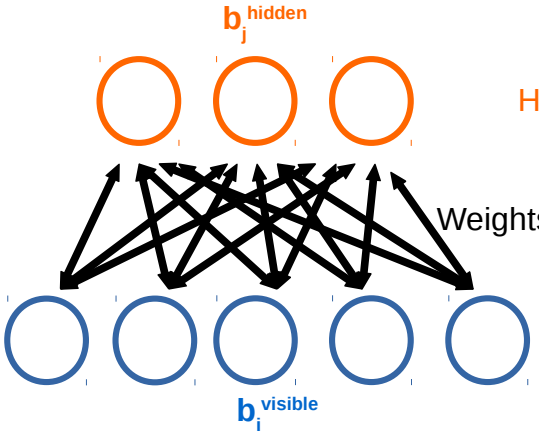
$$P(v,h) = e^{-E(v,h)} / Z$$

$$E(v,h) = \sum_{ij} W_{ij} v_i h_j + \text{bias terms}$$

Smolensky 1986; Teh and Hinton 2001; Hinton and Salakhutdinov 2006; Hinton 2007; Larochelle and Bengio 2008

Restricted Boltzmann Machines (RBMs)

Two-layer network



Probabilistic model of the joint distribution of **v** and **h** based on an **energy** function

Z partition function intractable

$$P(v,h) = e^{-E(v,h)} / Z$$

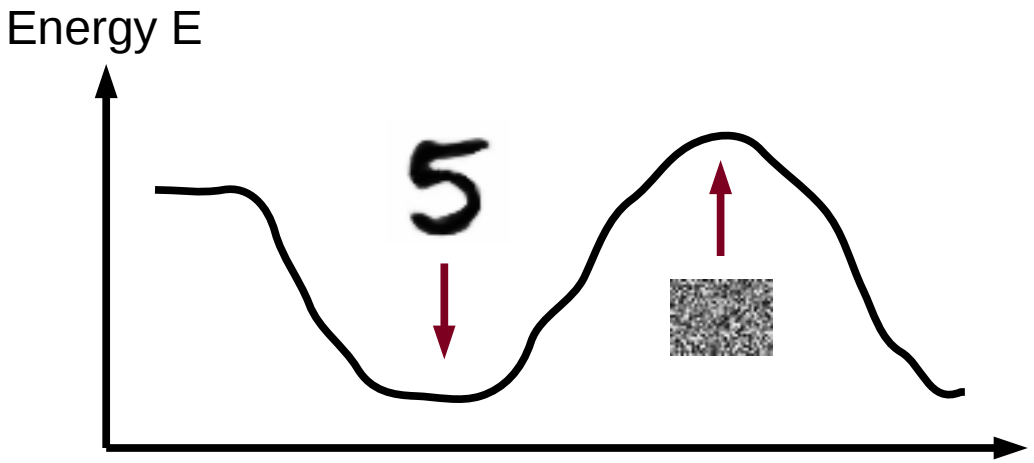
$$E(v,h) = \sum_{ij} W_{ij} v_i h_j + \text{bias terms}$$

Z intractable but we can sample from the conditional distribution P(h|x) and P(x|h)

A Contrastive Divergence algorithm gives an approximation of P(v,h)

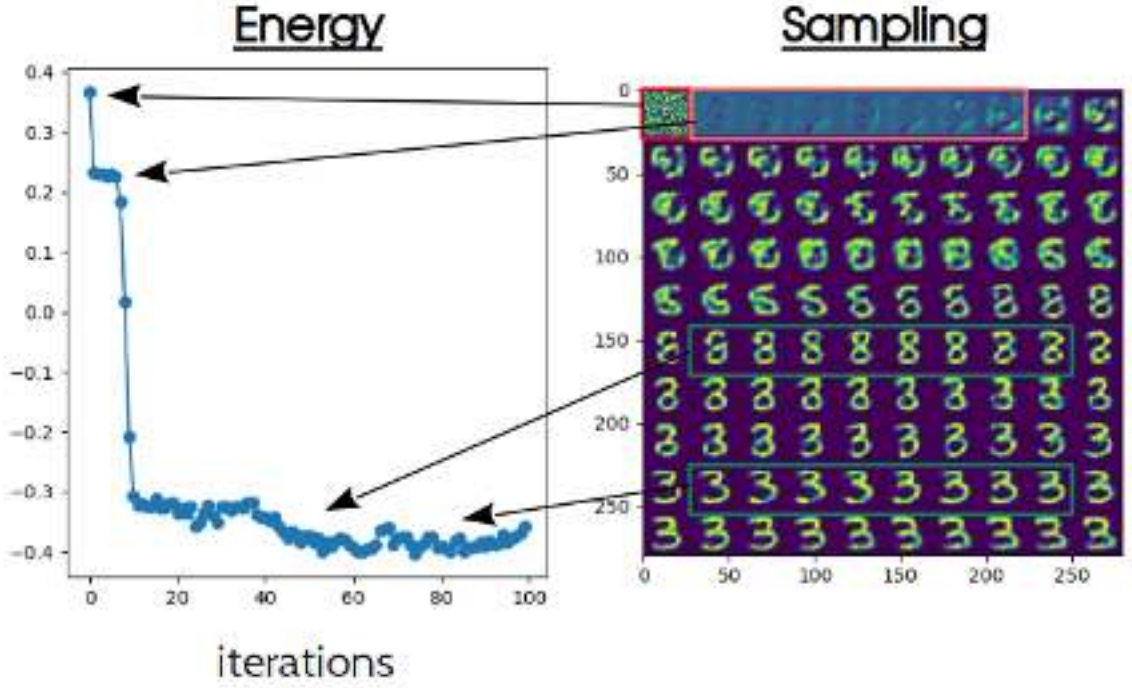
RBM

Contrastive Divergence algorithm gives an approximation of P(v,h)



Intuition:
- minimize the Energy of real examples (eg v(t), or similar example)
- maximize the Energy of the rest (or instead of a negative example v(t)*, i.e. point generated from random h)
until convergence

Restricted Boltzmann Machines (RBMs)



Intuition: **minimize the Energy of real examples** (or similar examples), and **maximize the energy of the rest** (points generated from random h) **until convergence**

Our implementation of RBM

100 or 500 hidden nodes

Sigmoid or ReLU activation functions

k=10 or 100 in Persistent CD-k

Learning rate between 0.001 and 0.0001

AG project overview

- 2 Generative Models
  - Generative Adversarial Networks
  - Restricted Boltzmann Machines

- **Create synthetic data from 1000Genomes (worldwide human populations data) and check preservation of data characteristics**

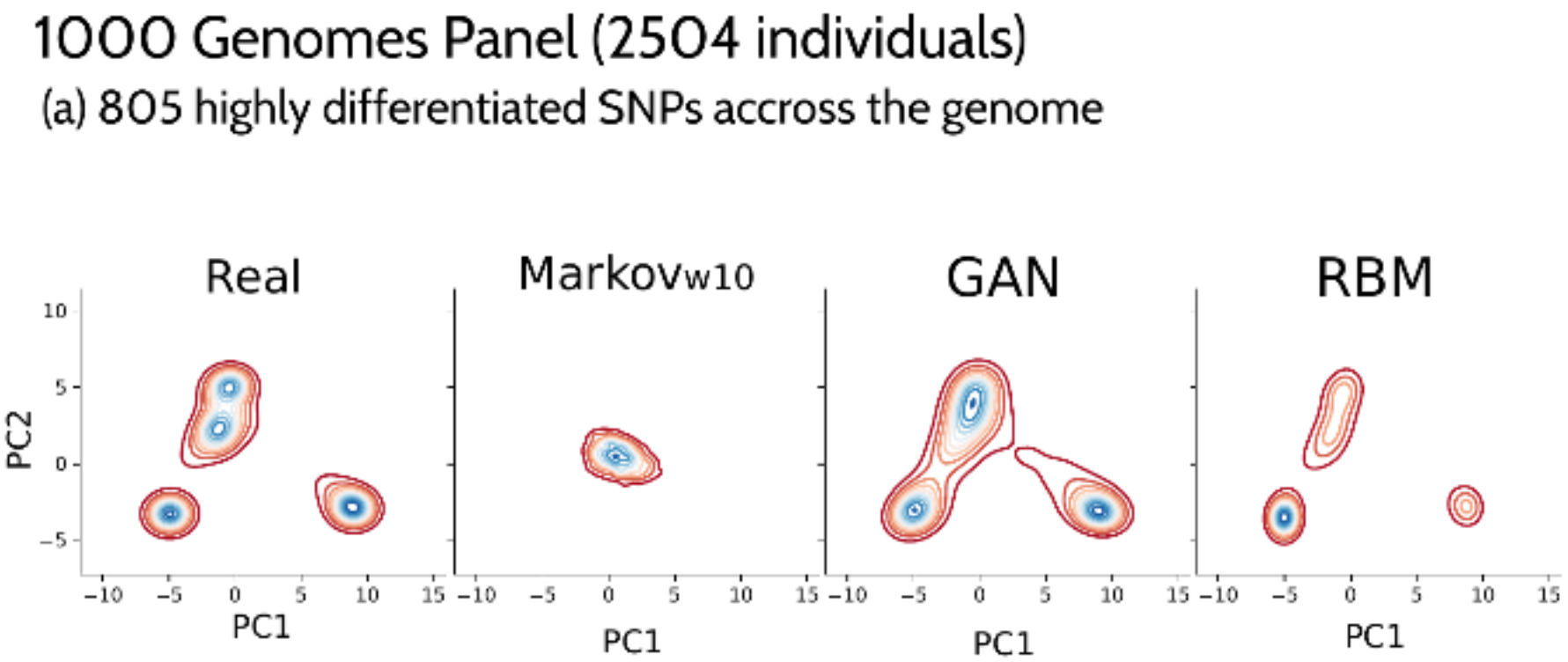
- Applications to the private Estonian biobank

- Statistics for overfitting and privacy loss detection

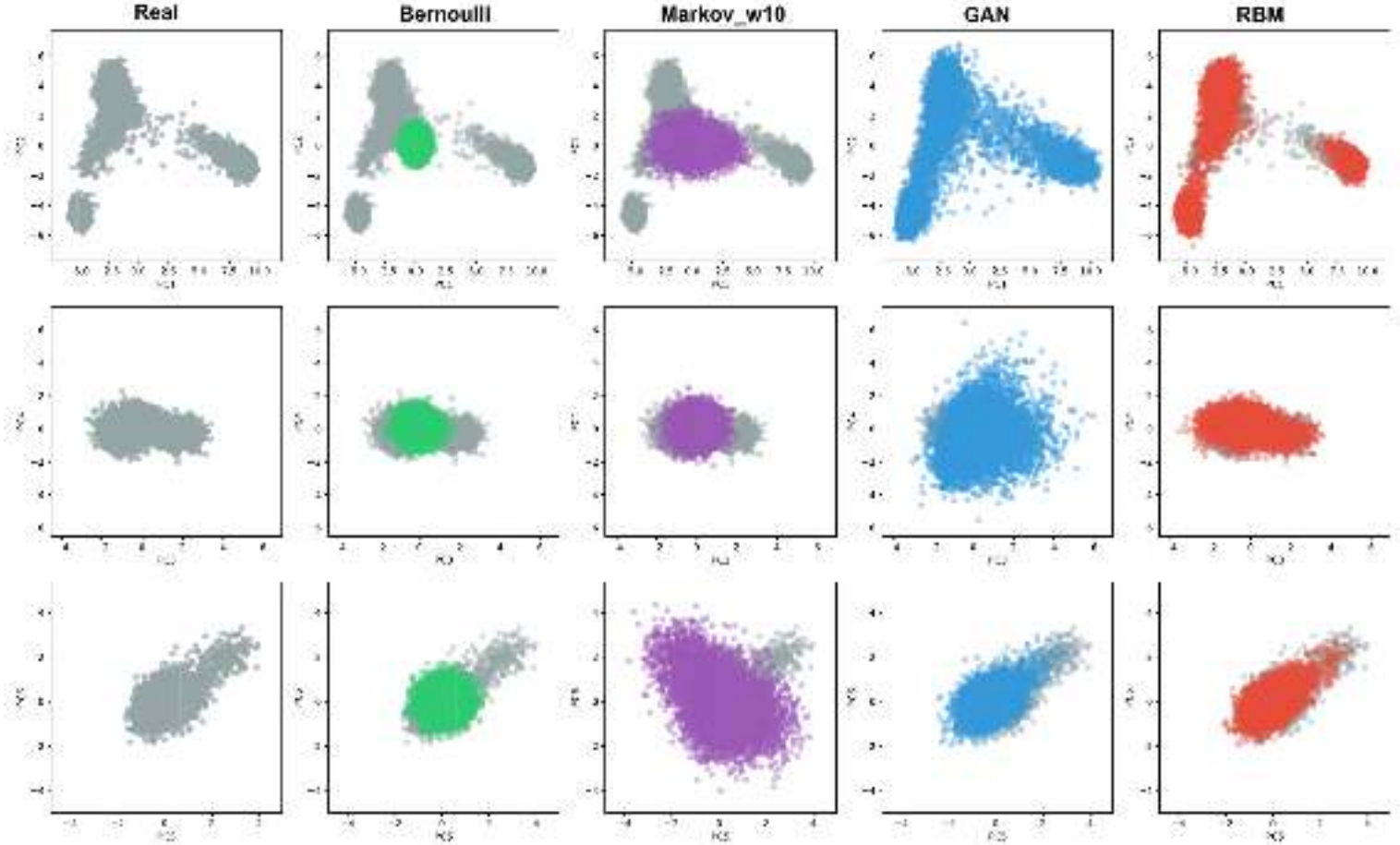
Are population genetic characteristics preserved?

Quality control (hidden population structure, ie relationships between indiv/lines)

PCA 2D density plots

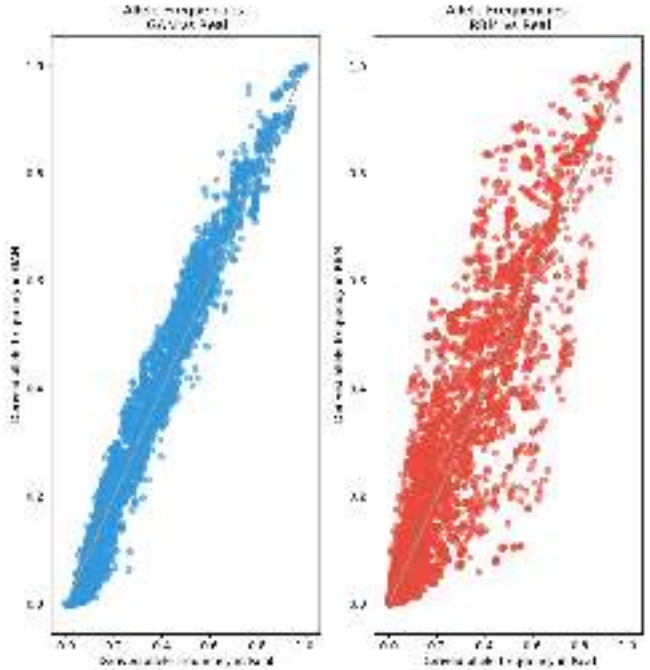


Quality control (hidden population structure, ie relationships between indiv/lines)

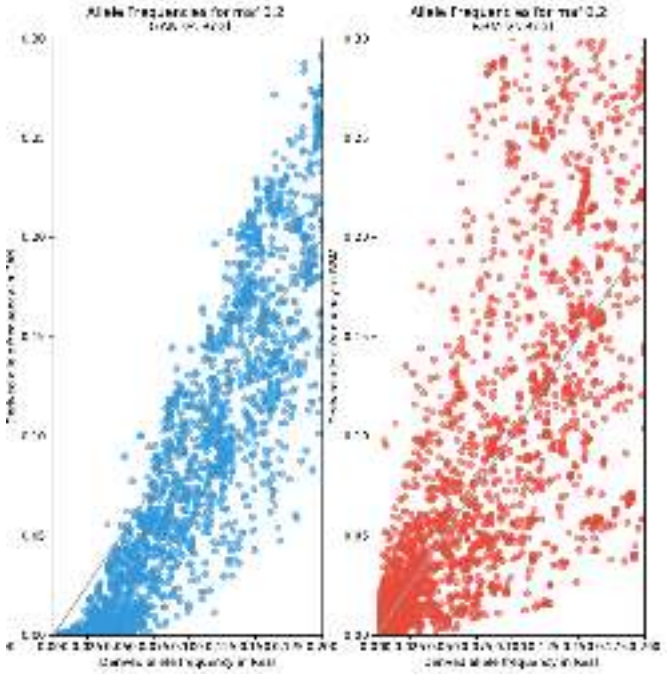


Quality control II. (allele frequency at each SNP, ie feature frequency)

10K successive SNP dataset

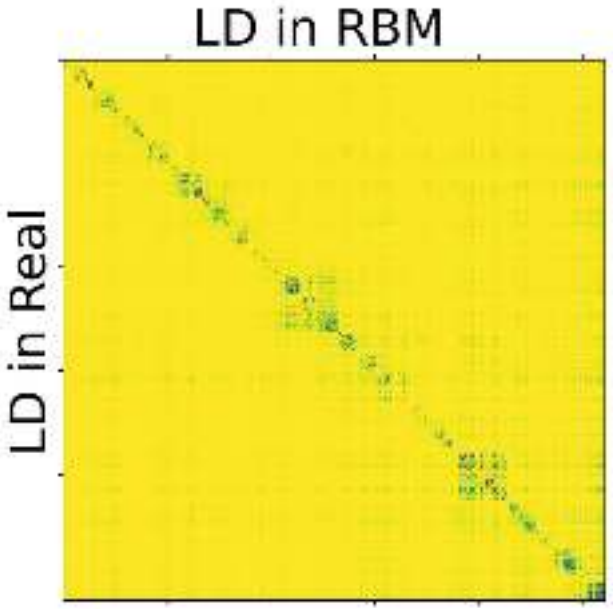


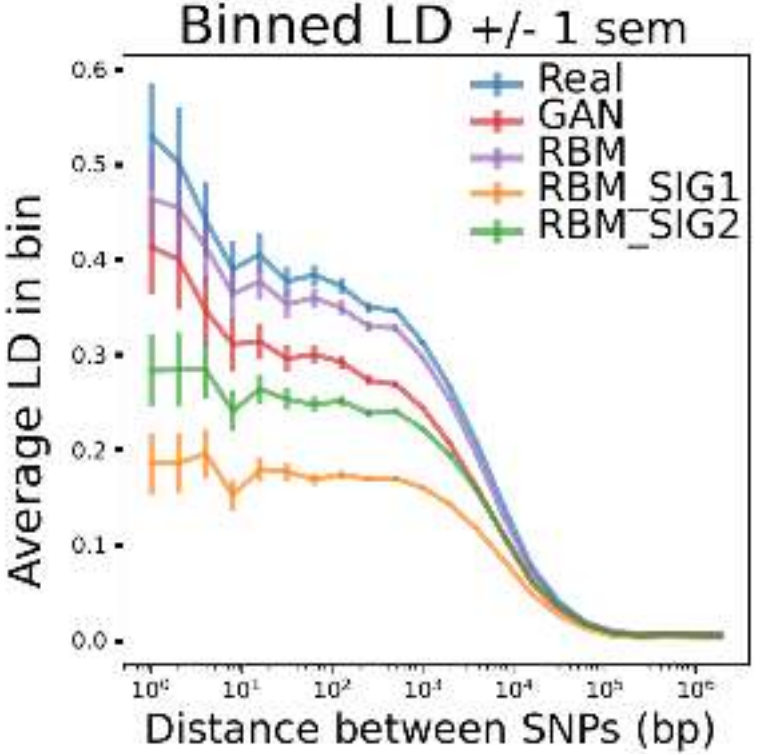
Zoom on low frequency features



Quality control III. (structure along the genome, ie correlation between columns)

(b) 10K consecutive SNPs

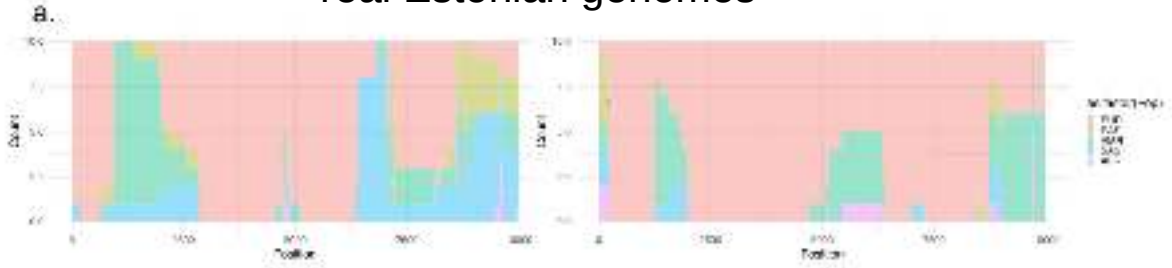




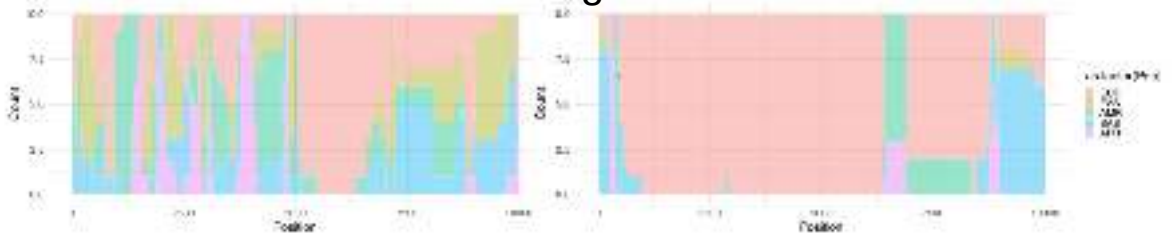
Quality control IV.

”Chromopainting” genomes using 1000 Genomes reference panel

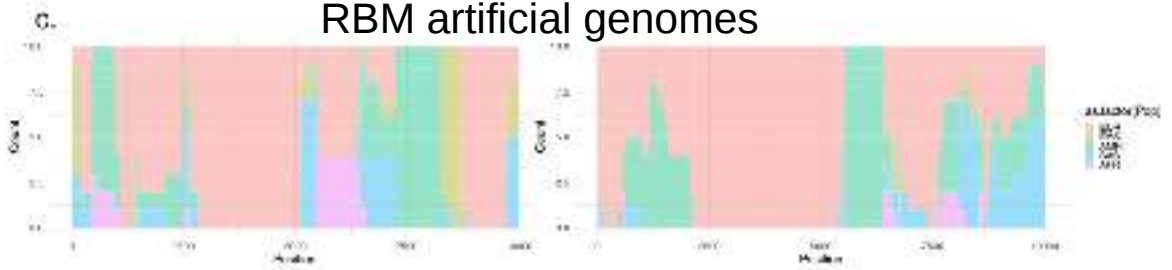
real Estonian genomes



GAN artificial genomes



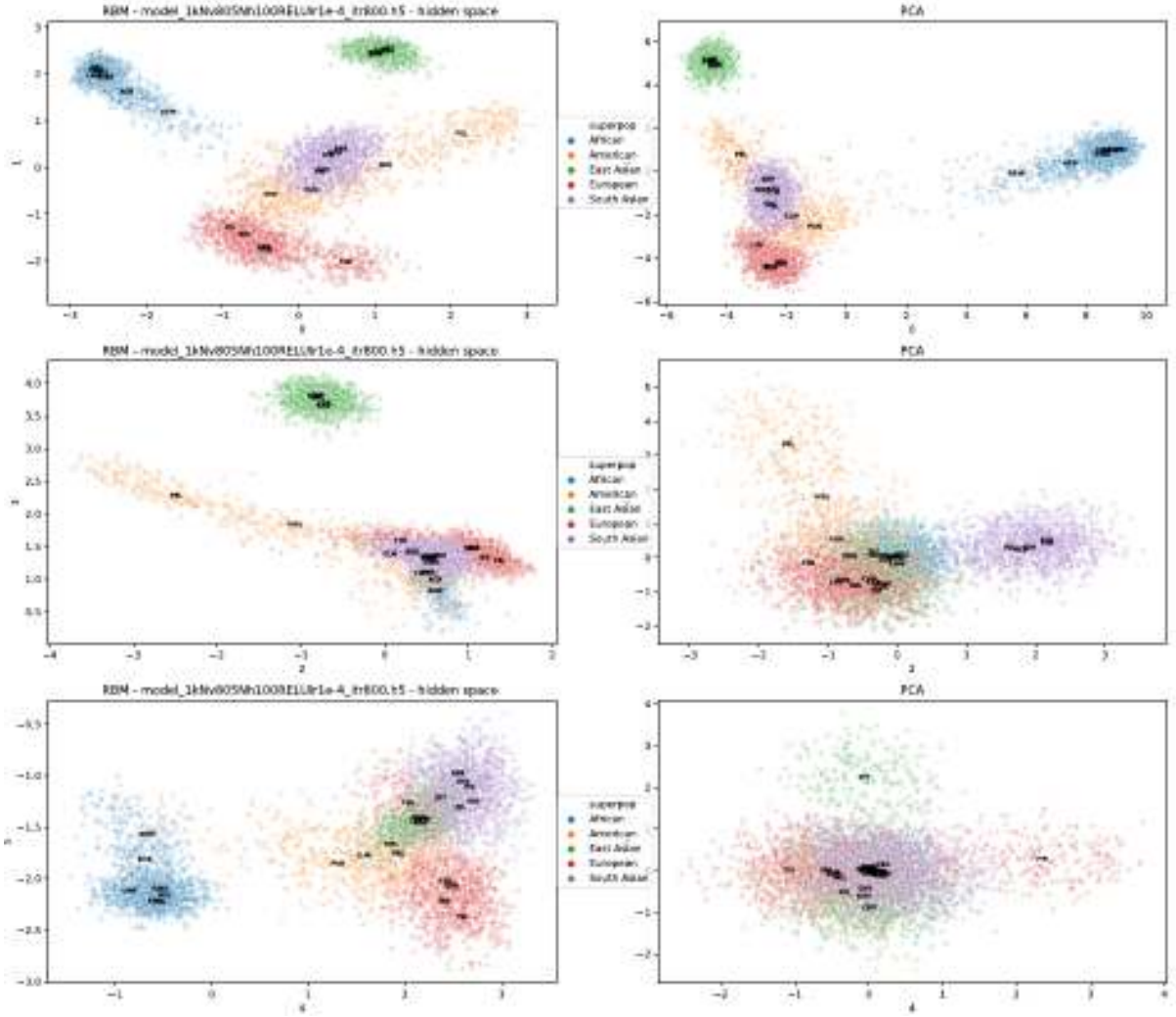
RBM artificial genomes



Application I. RBM Learned representation

Observing the hidden space **h** computed for **real** samples

→ a non linear alternative to PCA for dimension reduction

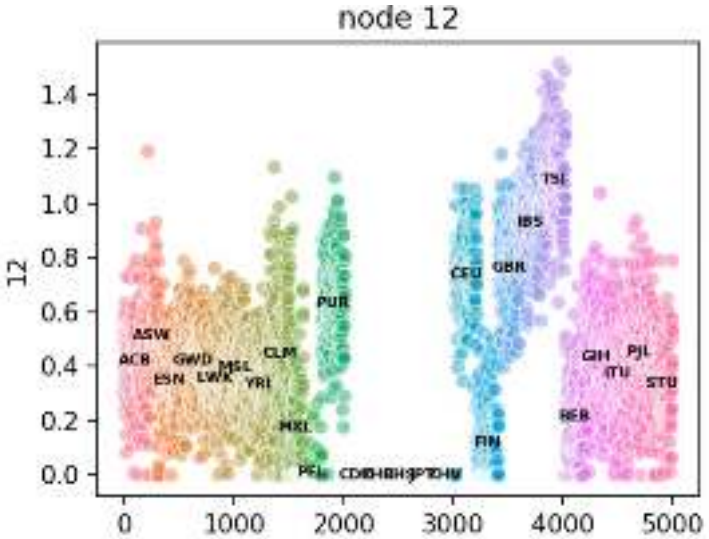


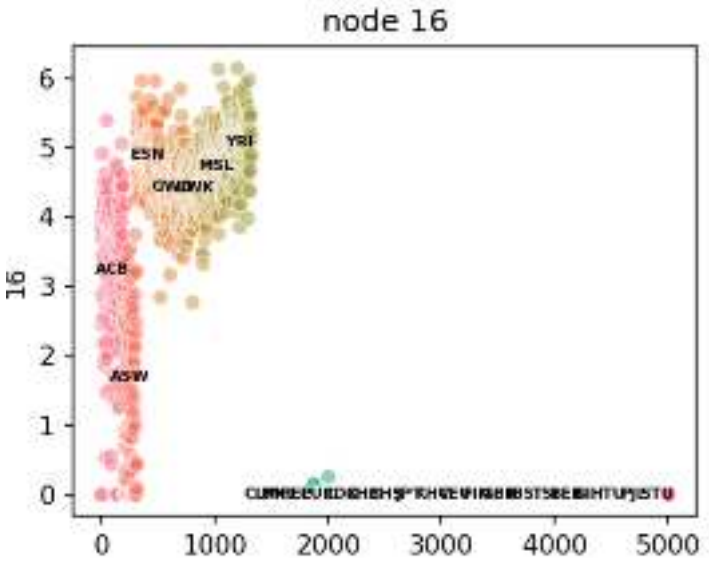
Application I. RBM learned representation

Observing the hidden space **h** computed for **real** samples

→ a non linear alternative to PCA for dimension reduction

→ check how hidden nodes are activated by real samples





AG project overview

- 2 Generative Models
  - Generative Adversarial Networks
  - Restricted Boltzmann Machines

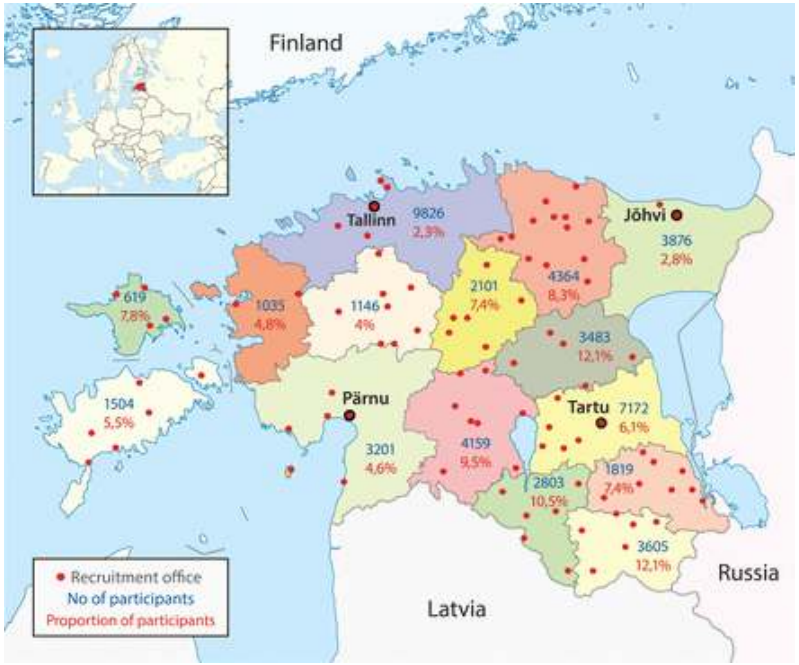
- Create synthetic data from 1000Genomes (worldwide human populations data) and check preservation of data characteristics

- **Applications to the private Estonian biobank**

- Statistics for overfitting and privacy loss detection

Application to the Estonian biobank

Private dataset, thousands of individuals, good quality (Leitsalu et al 2015)



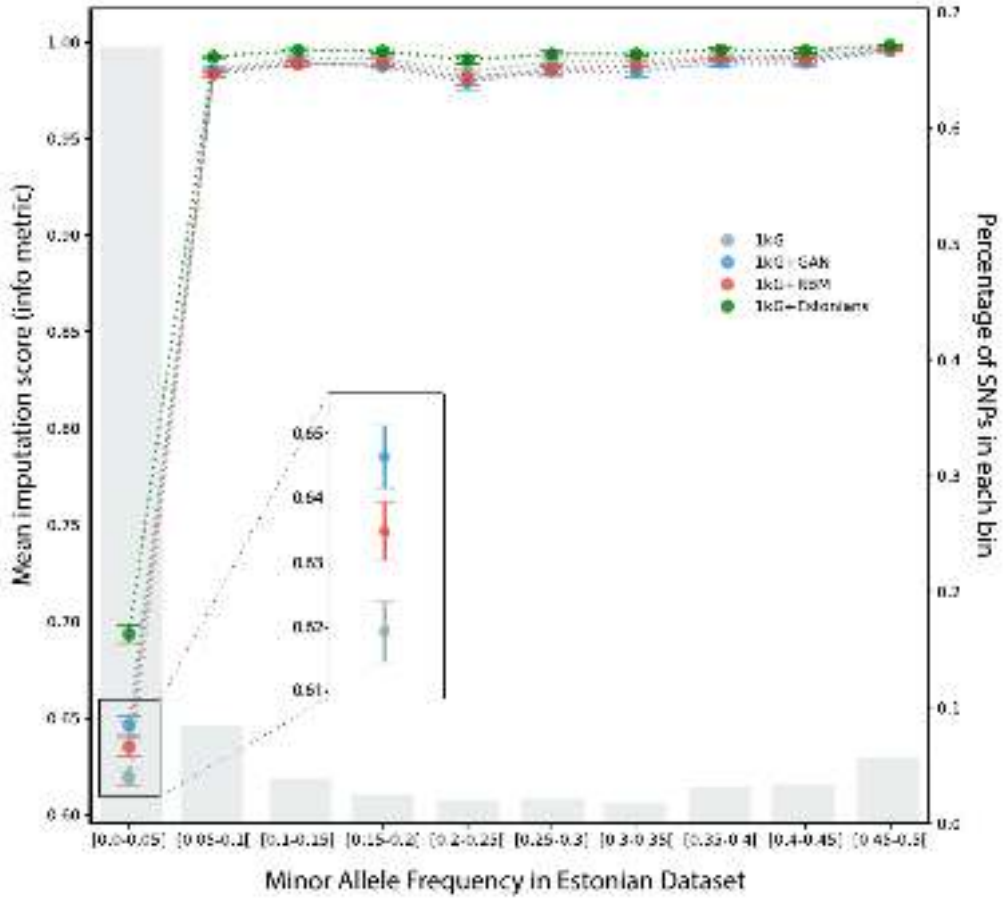
Application II.
Imputation of missing data

‘Perfect’ reference panel:
1000G + Estonian private dataset

‘Regular’ reference panel:
1000G

Suggested panel:
1000G + Artificial Genomes

→ improves performances for low frequency bin compare to the regular imputation scheme



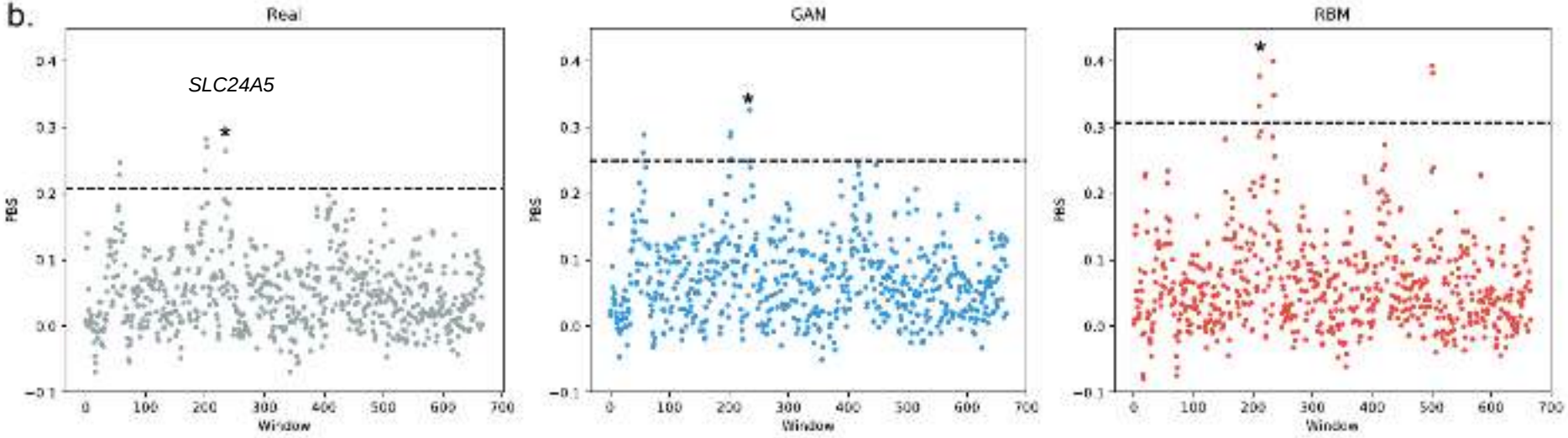
Application III. Detecting selection

3348 SNP region homogenously dispersed over chromosome 15

Population Branch Length (PBS): statistic for detecting selection based on 3-population trees (here Estonian, Yoruba, Japanese) [analysis also done with XP-EHH]

Good correlation between real and GAN PBS: 0.923; real and RBM PBS: 0.755

High peaks in real data also captured in AGs



---- = 99th percentile * = highest peaks

Yi et al. 2010, Sabeti et al. 2007

AG project overview

- 2 Generative Models
  - Generative Adversarial Networks
  - Restricted Boltzmann Machines
- Create synthetic data from 1000Genomes (worldwide human populations data) and check preservation of data characteristics
- Applications to the private Estonian biobank
- **Statistics for overfitting and privacy loss detection**

Detecting overfitting

Nearest Neighbour Adversarial Accuracy (AA_{TS})

Yale et al 2019

$$AA_{truth} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(d_{TS}(i) > d_{TT}(i))$$

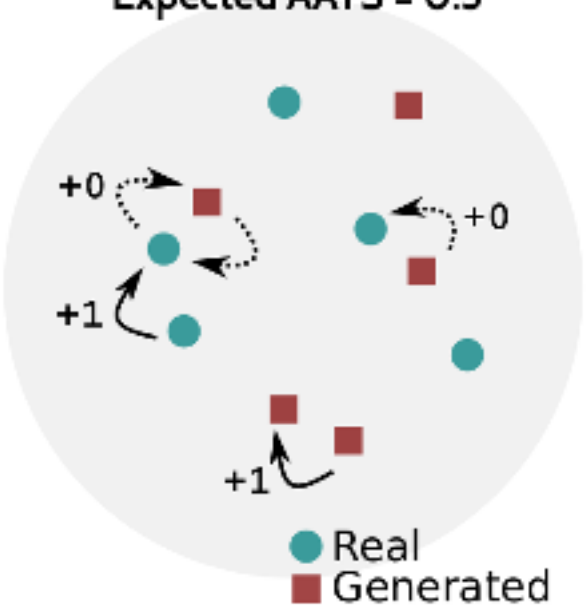
$$AA_{syn} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(d_{ST}(i) > d_{SS}(i))$$

$$AA_{TS} = \frac{1}{2}(AA_{true} + AA_{syn})$$

n = sample size
$d_{TS}(i)$ = distance between the real genome indexed by i
and its nearest neighbor in AG dataset
$d_{ST}(i)$ = distance between the artificial genome indexed by i
and its nearest neighbor in the real dataset
etc

AATS: sum[is the closest neighbor of each **Real** or **Fake** individual of the same type (+1) or not (+0) ?]

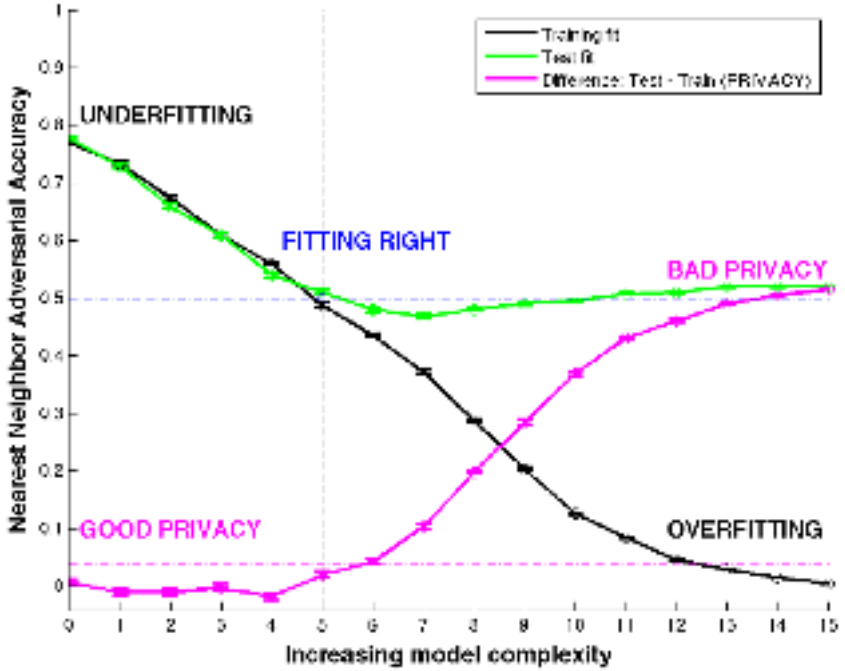
Expected AATS = 0.5



Detecting overfitting

Nearest Neighbour Adversarial Accuracy (AA_{TS})

Yale et al 2019

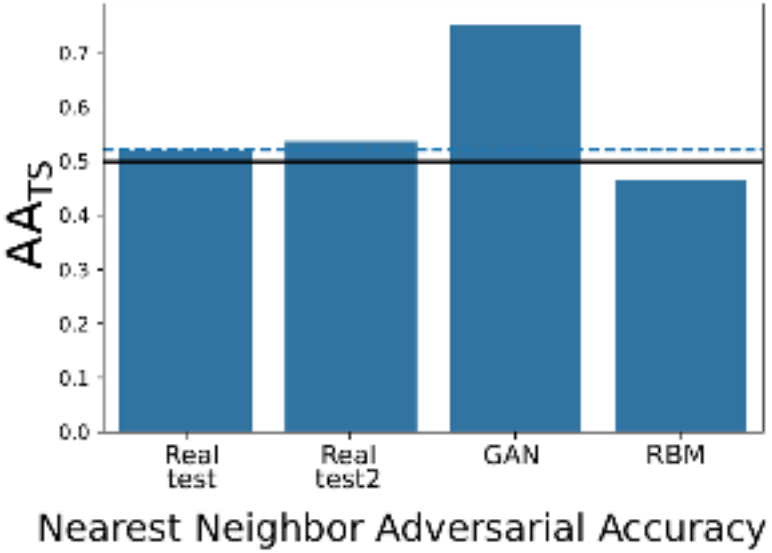


Example AA_{TS} scores

Estonian test sets -> AATS score is well calibrated (AA_{TS} ~ 0.5)

GAN is underfitting
RBM is overfitting

Estonian Biobank panel [4]
2k individuals for training,
4k for test.

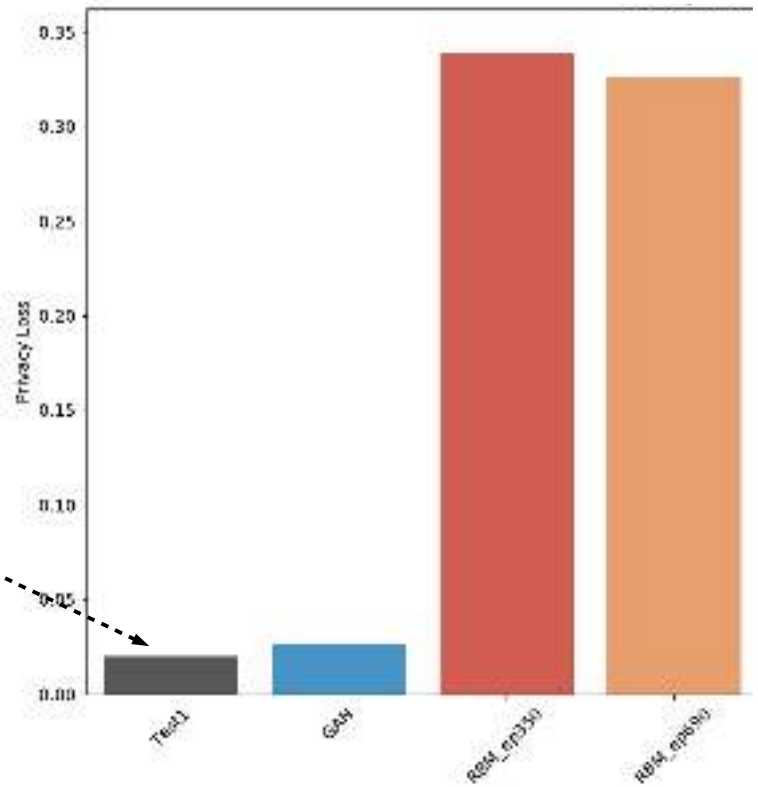


Privacy Loss for Estonian Dataset

Privacy Loss =

$$AA_{\text{Test Syn}} - AA_{\text{Train Syn}}$$

RBM is leaking information
GAN similar to releasing data never seen by the generative model

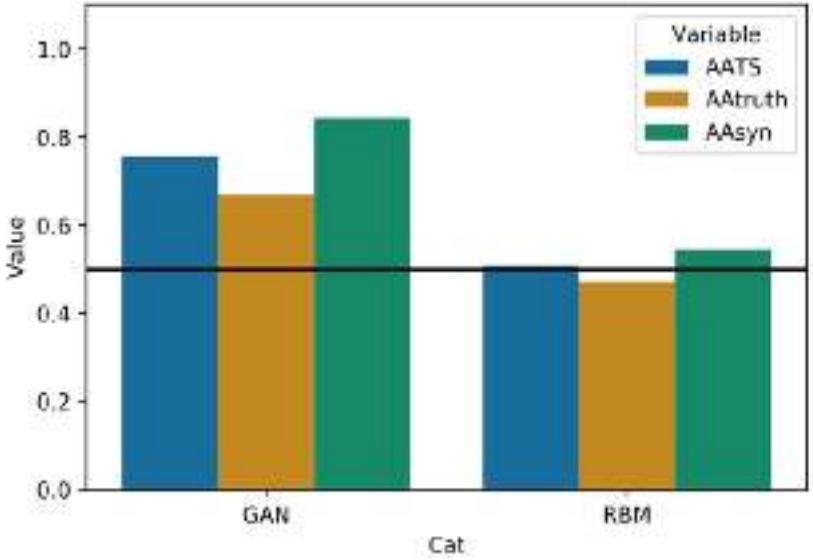


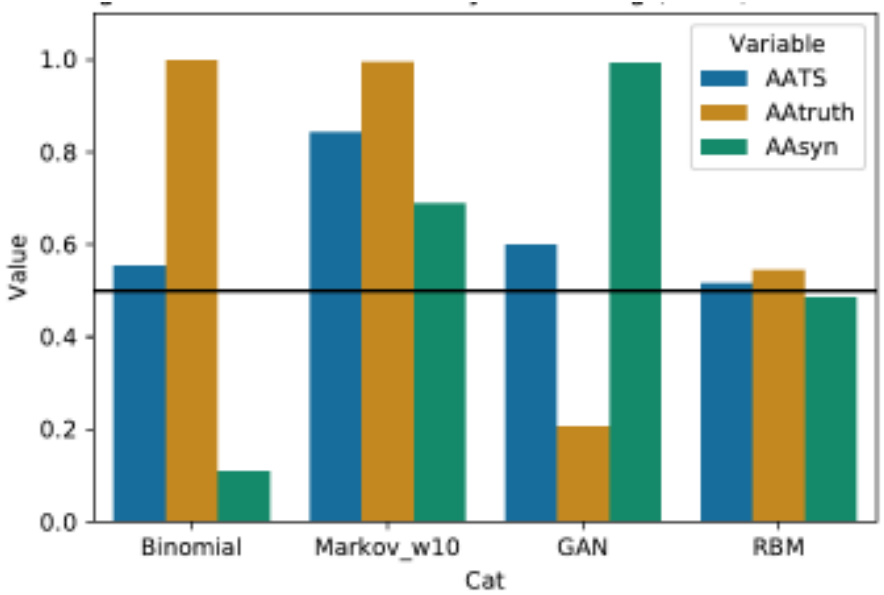
A new RBM sampling scheme to reduce Privacy Loss

Train RBM from real dataset (training set)
Generate new samples by starting the MCMC from another real dataset (sampling set)

Extension of AATS scores I.

Split the score into 2 parts
→ Observation of “correct” (left) versus “ill” (right) behaviour





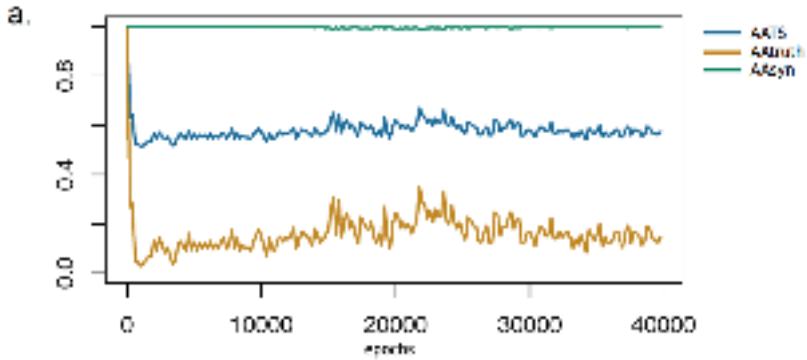
AATS over epochs

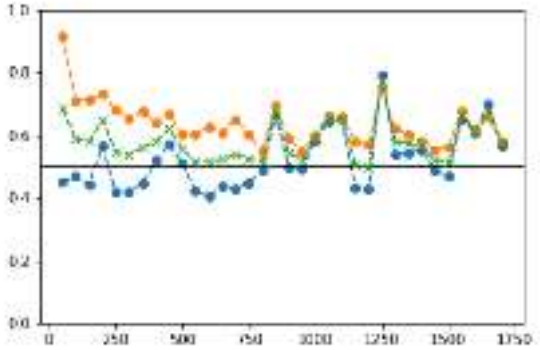
- Could be use as stopping criterion
- Sampling of AG to compute AA_{TS} is costly in RBMs

GAN

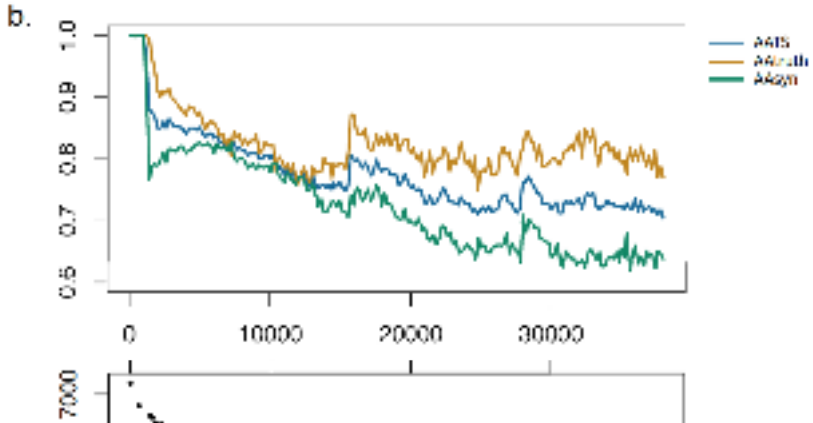
RBM

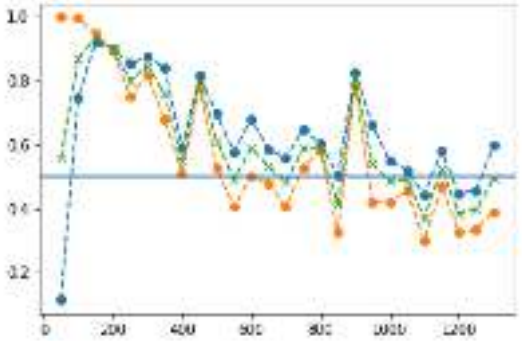
805 SNPs





10K SNPs



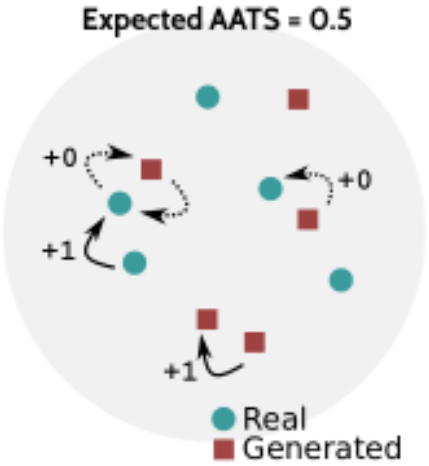


Extension of AATS II. up to 4-nearest neighbors

Model	AATS	TT	TS	ST	SS	TTT	TSS	STT	SSS	TTTT	TSSS	STTT	SSSS	TTTTT	TSSSS	STTTT	SSSSS
Ref. score	0.5	0.5	0.5	0.5	0.5	0.25	0.25	0.25	0.25	0.125	0.125	0.125	0.125	0.0625	0.0625	0.0625	0.0625
GAN	0.6177	0.2424	0.7576	0.007	0.993	0.1046	0.5974	0.0006	0.9806	0.0586	0.4736	0.0002	0.967	0.0372	0.3842	0	0.945
RBM	0.5503	0.6146	0.3854	0.514	0.486	0.3872	0.1542	0.269	0.2478	0.249	0.066	0.1434	0.1272	0.162	0.0292	0.0818	0.0638

Count various configurations such as TSS closest neigbhor is a Synthetic, 2nd closest also a Synthetic, etc

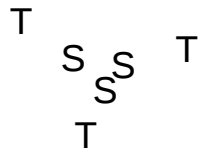
Expected relative frequencies for these configurations are known → compare to what is observed



Extension of AATS II. up to 4-nearest neighbors

Model	AATS	TT	TS	ST	SS	TTT	TSS	STT	SSS	TTTT	TSSS	STTT	SSSS	TTTTT	TSSSS	STTTT	SSSSS
Rel. score	0.5	0.5	0.5	0.5	0.5	0.25	0.25	0.25	0.25	0.125	0.125	0.125	0.125	0.0625	0.0625	0.0625	0.0625
GAN	0.6177	0.2424	0.7576	0.007	0.993	0.1046	0.5974	0.0006	0.9805	0.0586	0.4736	0.0002	0.967	0.0372	0.3842	0	0.945
RBM	0.3503	0.6146	0.3854	0.514	0.486	0.3872	0.1542	0.269	0.2478	0.249	0.066	0.1434	0.1272	0.162	0.0292	0.0818	0.0638

GAN for the 805 SNPs dataset has SS and **SSS** in high frequency
but TS and **TSS** also in high frequency
→ generating groups of Synthetic points clustered in the middle of True samples ?



→ **Extended AATS allows to identify complex cases of overfitting**

Summary AG project

- Unsupervised training of **GANs and RBMs** on (public and) **private datasets** to create **Artificial Genomes** with the idea of **augmenting public datasets**

- Proof-of-concept for **different type of genomic data**
  (dense short region of the genome, widespread with no physical linkage but strong population structure, SNPs uniformly sampled on a chromosome, genotype+phenotype, …)

- Difficulty and importance of **quality control** in a new area, extension of Adversarial Accuracy score

Summary AG project

- Unsupervised training of **GANs and RBMs** on (public and) **private** datasets to create **Artificial Genomes** with the idea of augmenting public datasets

- Proof-of-concept for **different type of data**
  (dense short region of the genome, widespread with no physical linkage but strong population structure, SNPs uniformly sampled on a chromosome, genotype+phenotype, …)

- Difficulty and importance of **quality control** in a new area, extension of AA score

- **Promising applications:**
  imputation, selection scan, GWAS (genotype/phenotype association), ...
  still many others could be tested

Summary AG project

- Unsupervised training of **GANs and RBMs** on (public and) **private** datasets to create **Artificial Genomes** with the idea of augmenting public datasets

- Proof-of-concept for **different type of data**
  (dense short region of the genome, widespread with no physical linkage but strong population structure, SNPs uniformly sampled on a chromosome, genotype+phenotype, …)

- Difficulty and importance of **quality control** in a new area, extension of AA score

- **Promising applications**: imputation, selection scan, GWAS (genotype/phenotype association), still many others could be tested

Next steps

- Extend to full genomes, a computational/architectural challenge
- What is the notion of privacy in population genetics and can we really ensure it ?
- Leverage generative/autoencoding abilities for inference tasks

Acknowledgments

- Inria TAU team for providing computational resources (GPU)
- All co-autors and in particular **Burak Yelmen and Théophile Sanchez**
- Adrien Pavao and other members of TAU (LRI)

- *Fundings*
  European Union through the European Regional Development Fund
  Estonian Research Council grant
  Laboratoire de Recherche en Informatique
  Center for Data Science

- *Inspiration for preparing these slides: F Li, I Goodfellow, H Larochelle and A Ghodse's lectures*